
GDK Documentation

Release 0.0.65

Blockstream Corporation

Jul 28, 2023

Contents:

1	GDK Functions	1
2	GDK JSON	25
3	GDK Create Transaction JSON	91
4	GDK Notifications	95
5	GDK Hardware Wallet Interface	99
6	GDK Swap Interface	105
7	LiquiDEX	107
8	Indices and tables	109
	Index	111

int **GA_init** (const GA_json* *config*)

Perform one-time initialization of the library. This call must be made once only before calling any other GDK functions, including any functions called from other threads.

Parameters

- **config** – The *Initialization config JSON*.

Returns GA_OK or an error code.

Return type int

int **GA_get_thread_error_details** (GA_json** *output*)

Get any error details associated with the last error on the current thread.

Parameters

- **output** – Destination for the output *Error details JSON* JSON. Returned GA_json should be freed using *GA_destroy_json*.

Returns GA_OK or an error code.

Return type int

int **GA_create_session** (struct GA_session** *session*)

Create a new session.

Parameters

- **session** – Destination for the resulting session. The returned session should be freed using *GA_destroy_session*.

Once created, the caller should set a handler for notifications using *GA_set_notification_handler*, before calling *GA_connect* to connect the session to the network for use. :return: GA_OK or an error code.
:rtype: int

int **GA_set_notification_handler** (struct GA_session* *session*, GA_notification_handler *handler*, void* *context*)

Set a handler to be called when notifications arrive for a session.

Parameters

- **session** – The session to receive notifications for.
- **handler** – The handler to receive notifications.
- **context** – A context pointer to be passed to the handler.

This call must be initially made on a session before *GA_connect*. *GDK Notifications* may arrive on different threads, so the caller must ensure that shared data is correctly locked within the handler. The *GA_json* object passed to the caller must be destroyed by the caller using *GA_destroy_json*. Failing to do so will result in memory leaks.

Once a session has been connected, this call can be made only with null values for *handler* and *context*. Once this returns, no further notifications will be delivered for the lifetime of the session.

The caller should not call session functions from within the callback handler as this may block the application.
:return: GA_OK or an error code. :rtype: int

int **GA_destroy_session** (struct GA_session* *session*)
Free a session allocated by *GA_create_session*.

Parameters

- **session** – The session to free.

If the session was connected using *GA_connect* then this call will disconnect it before destroying it. :return: GA_OK or an error code. :rtype: int

int **GA_connect** (struct GA_session* *session*, const GA_json* *net_params*)
Connect the session to the specified network.

Parameters

- **session** – The session to connect.
- **net_params** – The *Connection parameters JSON* of the network to connect to.

This call connects to the remote network services that the session requires, for example the Green servers or Electrum servers. *GA_connect* must be called only once per session lifetime, after *GA_create_session* and before *GA_destroy_session* respectively. Once connected, the underlying network connection of the session can be controlled using *GA_reconnect_hint*.

Once the session is connected, use *GA_register_user* to create a new wallet for the session, or *GA_login_user* to open an existing wallet. :return: GA_OK or an error code. :rtype: int

int **GA_reconnect_hint** (struct GA_session* *session*, const GA_json* *hint*)
Connect or disconnect a sessions underlying network connection.

Parameters

- **session** – The session to use.
- **hint** – the *Reconnect JSON* describing the desired reconnection behaviour.

Returns GA_OK or an error code.

Return type int

int **GA_get_proxy_settings** (struct GA_session* *session*, GA_json** *output*)
Get the current proxy settings for the given session.

Parameters

- **session** – The session to use.

- **output** – Destination for the output *Proxy Settings JSON*. Returned GA_json should be freed using *GA_destroy_json*.

Returns GA_OK or an error code.

Return type int

int **GA_get_wallet_identifier** (const GA_json* *net_params*, const GA_json* *params*, GA_json** *output*)

Compute a hashed wallet identifier from a BIP32 xpub or mnemonic.

The identifier returned is computed from the network combined with the master chain code and public key of the xpub/mnemonic. It can be used as a unique wallet identifier to mitigate privacy risks associated with storing the wallet's xpub.

Parameters

- **net_params** – The *Connection parameters JSON* of the network to compute an identifier for.
- **params** – The *Wallet identifier request JSON* to compute an identifier for.
- **output** – Destination for the output JSON. Returned GA_json should be freed using *GA_destroy_json*.

Returns GA_OK or an error code.

Return type int

int **GA_http_request** (struct GA_session* *session*, const GA_json* *params*, GA_json** *output*)

Make a request to an http server.

Parameters

- **session** – The session to use.
- **params** – the *HTTP parameters JSON* of the server to connect to.
- **output** – Destination for the output JSON. Returned GA_json should be freed using *GA_destroy_json*.

Returns GA_OK or an error code.

Return type int

int **GA_refresh_assets** (struct GA_session* *session*, const GA_json* *params*)

Refresh the sessions internal cache of Liquid asset information.

Each release of GDK comes with a partial list of Liquid assets built-in. This call is used to update it to include all the registered Liquid assets or any new assets that have been registered since the last update.

Parameters

- **session** – The session to use.
- **params** – the *Asset parameters JSON* of the server to connect to.

Returns GA_OK or an error code.

Return type int

int **GA_get_assets** (struct GA_session* *session*, const GA_json* *params*, GA_json** *output*)

Query the Liquid asset registry.

This call is used to retrieve informations about a set of Liquid assets specified by their asset id.

Parameters

- **session** – The session to use.
- **params** – the *Get assets parameters JSON* specifying the assets to query.
- **output** – Destination for the output *Asset details JSON*. Returned GA_json should be freed using *GA_destroy_json*.

Returns GA_OK or an error code.

Return type int

int **GA_validate_asset_domain_name** (struct GA_session* session, const GA_json* params,
GA_json** output)

Validate asset domain name. (This is a interface stub)

Returns GA_OK or an error code.

Return type int

int **GA_validate** (struct GA_session* session, GA_json* details, struct GA_auth_handler** call)

Validate a gdk format JSON document.

Parameters

- **session** – The session to use.
- **details** – The *Validate JSON* to validate.
- **call** – Destination for the resulting GA_auth_handler to complete the action. The call handlers result is *Validate Result JSON*. Returned GA_auth_handler should be freed using *GA_destroy_auth_handler*.

Note: When calling from C/C++, the parameter details will be emptied when the call

completes. :return: GA_OK or an error code. :rtype: int

int **GA_register_user** (struct GA_session* session, GA_json* hw_device, GA_json* details, struct
GA_auth_handler** call)

Create a new user wallet.

Parameters

- **session** – The session to use.
- **hw_device** – *HW device JSON* or empty JSON for software wallet registration.
- **details** – The *Login credentials JSON* for software wallet registration.
- **call** – Destination for the resulting GA_auth_handler to perform the registration. Returned GA_auth_handler should be freed using *GA_destroy_auth_handler*.

Note: When calling from C/C++, the parameters hw_device and details will be emptied when the call

completes. :return: GA_OK or an error code. :rtype: int

int **GA_login_user** (struct GA_session* session, GA_json* hw_device, GA_json* details, struct
GA_auth_handler** call)

Authenticate to a user's wallet.

Parameters

- **session** – The session to use.
- **hw_device** – *HW device JSON* or empty JSON for software wallet login.

- **details** – The *Login credentials JSON* for authenticating the user.
- **call** – Destination for the resulting GA_auth_handler to perform the login. Returned GA_auth_handler should be freed using *GA_destroy_auth_handler*.

If a sessions underlying network connection has disconnected and reconnected, the user will need to login again using this function. In this case, the caller can pass empty JSON for both hw_device and details to login using the previously passed credentials and device.

Note: When calling from C/C++, the parameters hw_device and details will be emptied when the call

completes. :return: GA_OK or an error code. :rtype: int

int **GA_set_watch_only** (struct GA_session* session, const char* username, const char* password)

Set or disable a watch-only login for a logged-in user wallet.

Parameters

- **session** – The session to use.
- **username** – The watch-only username to login with, or a blank string to disable.
- **password** – The watch-only password to login with, or a blank string to disable.

Returns GA_OK or an error code.

Return type int

int **GA_get_watch_only_username** (struct GA_session* session, char** username)

Get the current watch-only login for a logged-in user wallet, if any.

Parameters

- **session** – The session to use.
- **username** – Destination for the watch-only username. Empty string if not set. Returned string should be freed using *GA_destroy_string*.

Returns GA_OK or an error code.

Return type int

int **GA_remove_account** (struct GA_session* session, struct GA_auth_handler** call)

Remove and delete the server history of a wallet.

Parameters

- **session** – The session to use.
- **call** – Destination for the resulting GA_auth_handler to perform the removal. Returned GA_auth_handler should be freed using *GA_destroy_auth_handler*.

For multisig Green sessions, removing a wallet removes all history and data associated with the wallet on the server. This operation cannot be undone, and re-registering the wallet will not bring back the wallet's history. For this reason, only empty wallets can be deleted.

For singlesig sessions, removing a wallet removes the locally persisted cache. The actual removal will happen after *GA_destroy_session* is called. :return: GA_OK or an error code. :rtype: int

int **GA_create_subaccount** (struct GA_session* session, GA_json* details, struct GA_auth_handler** call)

Create a subaccount.

Parameters

- **session** – The session to use.
- **details** – The subaccount "name" (which must not be already used in the wallet) and "type" (either "2of2", "2of2_no_recovery" or "2of3") must be populated. Type "2of2_no_recovery" is available only for Liquid networks and always requires both keys for spending. For type "2of3" the caller may provide either "recovery_mnemonic" or "recovery_xpub" if they do not wish to have a mnemonic passphrase generated automatically. All other fields are ignored.
- **call** – Destination for the resulting GA_auth_handler to perform the creation. Returned GA_auth_handler should be freed using [GA_destroy_auth_handler](#). Details of the created subaccount are returned in the "result" element of the GA_auth_handler. For 2of3 subaccounts the field "recovery_xpub" will be populated, and "recovery_mnemonic" will contain the recovery mnemonic passphrase if one was generated. These values must be stored safely by the caller as they will not be returned again by any call such as [GA_get_subaccounts](#).

Note: When calling from C/C++, the parameter `details` will be emptied when the call completes.

Returns GA_OK or an error code.

Return type int

```
int GA_get_subaccounts (struct GA_session* session, const GA_json* details, struct
                        GA_auth_handler** call)
    Get the user's subaccount details.
```

Parameters

- **session** – The session to use.
- **details** – the [Get Subaccounts parameters JSON](#) controlling the request.
- **call** – Destination for the resulting GA_auth_handler to perform the creation. The call handlers result is [Subaccounts list JSON](#). Returned GA_auth_handler should be freed using [GA_destroy_auth_handler](#).

Returns GA_OK or an error code.

Return type int

```
int GA_get_subaccount (struct GA_session* session, uint32_t subaccount, struct
                      GA_auth_handler** call)
    Get subaccount details.
```

Parameters

- **session** – The session to use.
- **subaccount** – The value of "pointer" from [Subaccounts list JSON](#) for the subaccount.
- **call** – Destination for the resulting GA_auth_handler to perform the creation. The call handlers result is [Subaccount JSON](#). Returned GA_auth_handler should be freed using [GA_destroy_auth_handler](#).

Returns GA_OK or an error code.

Return type int

```
int GA_rename_subaccount (struct GA_session* session, uint32_t subaccount, const char* new_name)
    Rename a subaccount.
```

Parameters

- **session** – The session to use.
- **subaccount** – The value of "pointer" from *Subaccounts list JSON* or *Subaccount JSON* for the subaccount to rename.
- **new_name** – New name for the subaccount.

Note: This call is deprecated and will be removed in a future release. Use *GA_update_subaccount* to rename subaccounts.

Returns GA_OK or an error code.

Return type int

```
int GA_update_subaccount (struct GA_session* session, GA_json* details, struct
                        GA_auth_handler** call)
Update subaccount information.
```

Parameters

- **session** – The session to use.
- **details** – *Subaccount update JSON* giving the details to update.
- **call** – Destination for the resulting GA_auth_handler to complete the action. Returned GA_auth_handler should be freed using *GA_destroy_auth_handler*.

Note: When calling from C/C++, the parameter *details* will be emptied when the call completes

Returns GA_OK or an error code.

Return type int

```
int GA_get_transactions (struct GA_session* session, GA_json* details, struct
                        GA_auth_handler** call)
Get a page of the user's transaction history.
```

Parameters

- **session** – The session to use.
- **details** – *Transactions details JSON* giving the details to get the transactions for.
- **call** – Destination for the resulting GA_auth_handler to complete the action. Returned GA_auth_handler should be freed using *GA_destroy_auth_handler*.

Note: When calling from C/C++, the parameter *details* will be emptied when the call completes.

Note: Transactions are returned as *Transaction list JSON* from newest to oldest with up to 30 transactions per page.

Returns GA_OK or an error code.

Return type int

int **GA_get_receive_address** (struct GA_session* session, GA_json* details, struct GA_auth_handler** call)

Get a new address to receive coins to.

Parameters

- **session** – The session to use.
- **details** – *Receive address request JSON*.
- **call** – Destination for the resulting GA_auth_handler to complete the action. The call handlers result is *Receive address details JSON*. Returned GA_auth_handler should be freed using *GA_destroy_auth_handler*.

Note: When calling from C/C++, the parameter `details` will be emptied when the call completes.

Returns GA_OK or an error code.

Return type int

int **GA_get_previous_addresses** (struct GA_session* session, GA_json* details, struct GA_auth_handler** call)

Get a page of addresses previously generated for a subaccount.

Parameters

- **session** – The session to use.
- **details** – *Previous addresses request JSON* detailing the previous addresses to fetch.
- **call** – Destination for the resulting GA_auth_handler to complete the action. The call handlers result is *Previous addresses JSON*. Returned GA_auth_handler should be freed using *GA_destroy_auth_handler*.

Note: When calling from C/C++, the parameter `details` will be emptied when the call completes.

Note: Iteration of all addresses is complete when ‘last_pointer’ is not present in the results.

Returns GA_OK or an error code.

Return type int

int **GA_get_unspent_outputs** (struct GA_session* session, GA_json* details, struct GA_auth_handler** call)

Get the user’s unspent transaction outputs.

Parameters

- **session** – The session to use.
- **details** – *Unspent outputs request JSON* detailing the unspent transaction outputs to fetch.

- **call** – Destination for the resulting GA_auth_handler to complete the action. The call handlers result is *Unspent outputs JSON*. Returned GA_auth_handler should be freed using *GA_destroy_auth_handler*.

Note: When calling from C/C++, the parameter `details` will be emptied when the call completes.

Returns GA_OK or an error code.

Return type int

int **GA_get_unspent_outputs_for_private_key** (struct GA_session* session, GA_json* details, struct GA_auth_handler** call)

Get the unspent transaction outputs associated with a non-wallet private key.

Parameters

- **session** – The session to use.
- **details** – *Unspent outputs for private key request JSON* detailing the private key to check.
- **call** – Destination for the resulting GA_auth_handler to complete the action. The call handlers result is *Unspent outputs JSON*. Returned GA_auth_handler should be freed using *GA_destroy_auth_handler*.

Note: Neither the private key or its derived public key are sent to any third party for this call.

Returns GA_OK or an error code.

Return type int

int **GA_set_unspent_outputs_status** (struct GA_session* session, GA_json* details, struct GA_auth_handler** call)

Change the status of a user's unspent transaction outputs.

Parameters

- **session** – The session to use.
- **details** – *Unspent outputs set status JSON* detailing the unspent transaction outputs status to set.
- **call** – Destination for the resulting GA_auth_handler to complete the action. Returned GA_auth_handler should be freed using *GA_destroy_auth_handler*.

Note: When calling from C/C++, the parameter `details` will be emptied when the call completes.

Returns GA_OK or an error code.

Return type int

int **GA_get_transaction_details** (struct GA_session* session, const char* txhash_hex, GA_json** transaction)

Get a transaction's details.

Parameters

- **session** – The session to use.

- **txhash_hex** – The transaction hash of the transaction to fetch.
- **transaction** – Destination for the *Transaction details JSON*. Returned GA_json should be freed using *GA_destroy_json*.

Returns GA_OK or an error code.

Return type int

int **GA_get_balance** (struct GA_session* *session*, GA_json* *details*, struct GA_auth_handler** *call*)

Get the sum of unspent outputs paying to a subaccount.

Parameters

- **session** – The session to use.
- **details** – *Unspent outputs request JSON* detailing the unspent transaction outputs to compute the balance from.
- **call** – Destination for the resulting GA_auth_handler to complete the action. Returned GA_auth_handler should be freed using *GA_destroy_auth_handler*.

Note: When calling from C/C++, the parameter *details* will be emptied when the call completes.

Returns GA_OK or an error code.

Return type int

int **GA_get_available_currencies** (struct GA_session* *session*, GA_json** *currencies*)

Get the list of allowed currencies for all available pricing sources.

Parameters

- **session** – The session to use.
- **currencies** – The returned list of *Available currencies JSON*. Returned GA_json should be freed using *GA_destroy_json*.

Returns GA_OK or an error code.

Return type int

int **GA_convert_amount** (struct GA_session* *session*, const GA_json* *value_details*, GA_json** *output*)

Convert Fiat to BTC and vice-versa.

Parameters

- **session** – The session to use.
- **value_details** – *Convert amount JSON* giving the value to convert.
- **output** – Destination for the converted values *Amount JSON*. Returned GA_json should be freed using *GA_destroy_json*.

Returns GA_OK or an error code.

Return type int

int **GA_encrypt_with_pin** (struct GA_session* *session*, GA_json* *details*, struct GA_auth_handler** *call*)

Encrypt JSON with a server provided key protected by a PIN.

Parameters

- **session** – The session to use.

- **details** – The *Encrypt with PIN JSON* to encrypt.
- **call** – Destination for the resulting GA_auth_handler to complete the action. The call handlers result is *Encrypt with PIN Result JSON* which the caller should persist. Returned GA_auth_handler should be freed using *GA_destroy_auth_handler*.

Note: When calling from C/C++, the parameter `details` will be emptied when the call completes.

Returns GA_OK or an error code.

Return type int

int **GA_decrypt_with_pin** (struct GA_session* session, GA_json* details, struct GA_auth_handler** call)
 Decrypt JSON with a server provided key protected by a PIN.

Parameters

- **session** – The session to use.
- **details** – The *Decrypt with PIN JSON* to decrypt.
- **call** – Destination for the resulting GA_auth_handler to complete the action. The call handlers result is the decrypted JSON. Returned GA_auth_handler should be freed using *GA_destroy_auth_handler*.

Note: When calling from C/C++, the parameter `details` will be emptied when the call completes.

Returns GA_OK or an error code.

Return type int

int **GA_disable_all_pin_logins** (struct GA_session* session)
 Disable all PIN logins previously set.

After calling this method, the user will not be able to login with PIN from any device that was previously enabled using *GA_encrypt_with_pin*.

Parameters

- **session** – The session to use.

Returns GA_OK or an error code.

Return type int

int **GA_create_transaction** (struct GA_session* session, GA_json* transaction_details, struct GA_auth_handler** call)
 Construct a transaction.

Parameters

- **session** – The session to use.
- **transaction_details** – The *GDK Create Transaction JSON* for constructing.
- **call** – Destination for the resulting GA_auth_handler to complete the action. Returned GA_auth_handler should be freed using *GA_destroy_auth_handler*.

Note: When calling from C/C++, the parameter `transaction_details` will be emptied when the call completes.

Returns `GA_OK` or an error code.

Return type `int`

`int GA_blind_transaction (struct GA_session* session, GA_json* transaction_details, struct GA_auth_handler** call)`

Blind a transaction.

Parameters

- **session** – The session to use.
- **transaction_details** – The *GDK Create Transaction JSON* for blinding.
- **call** – Destination for the resulting `GA_auth_handler` to complete the action. Returned `GA_auth_handler` should be freed using *GA_destroy_auth_handler*.

Note: When calling from C/C++, the parameter `transaction_details` will be emptied when the call completes.

Returns `GA_OK` or an error code.

Return type `int`

`int GA_sign_transaction (struct GA_session* session, GA_json* transaction_details, struct GA_auth_handler** call)`

Sign the user's inputs to a transaction.

Parameters

- **session** – The session to use.
- **transaction_details** – The *Sign transaction JSON* for signing, as previously returned from *GA_create_transaction* or (for Liquid) *GA_blind_transaction*.
- **call** – Destination for the resulting `GA_auth_handler` to perform the signing. Returned `GA_auth_handler` should be freed using *GA_destroy_auth_handler*.

Note: When calling from C/C++, the parameter `transaction_details` will be emptied when the call completes.

Returns `GA_OK` or an error code.

Return type `int`

`int GA_create_swap_transaction (struct GA_session* session, const GA_json* swap_details, struct GA_auth_handler** call)`

Construct the initiators side of a swap transaction.

Parameters

- **session** – The session to use.
- **swap_details** – The *Create Swap Transaction JSON* for constructing.

- **call** – Destination for the resulting `GA_auth_handler` to complete the action. The call handlers result is *Create Swap Transaction Result JSON*. Returned `GA_auth_handler` should be freed using *GA_destroy_auth_handler*.

Returns `GA_OK` or an error code.

Return type `int`

`int GA_complete_swap_transaction (struct GA_session* session, const GA_json* swap_details, struct GA_auth_handler** call)`

Complete construction of the callers side of a swap transaction.

Parameters

- **session** – The session to use.
- **swap_details** – The *Complete Swap Transaction JSON* for completing.
- **call** – Destination for the resulting `GA_auth_handler` to complete the action. The call handlers result is *Complete Swap Transaction Result JSON*. Returned `GA_auth_handler` should be freed using *GA_destroy_auth_handler*.

Returns `GA_OK` or an error code.

Return type `int`

`int GA_psb_t_sign (struct GA_session* session, GA_json* details, struct GA_auth_handler** call)`

Sign one or more of a user's inputs in a PSBT or PSET.

Parameters

- **session** – The session to use.
- **details** – The *Sign PSBT JSON* for signing.
- **call** – Destination for the resulting `GA_auth_handler` to perform the signing. The call handlers result is *Sign PSBT Result JSON*. Returned `GA_auth_handler` should be freed using *GA_destroy_auth_handler*.

Note: When calling from C/C++, the parameter `details` will be emptied when the call completes.

Note: EXPERIMENTAL warning: this call may be changed in future releases.

Returns `GA_OK` or an error code.

Return type `int`

`int GA_psb_t_get_details (struct GA_session* session, GA_json* details, struct GA_auth_handler** call)`

Get wallet details of a PSBT or PSET.

Parameters

- **session** – The session to use.
- **details** – The *PSBT Get Details JSON* for getting the wallet details.
- **call** – Destination for the resulting `GA_auth_handler` to get the wallet details. The call handlers result is *PSBT Get Details Result JSON*. Returned `GA_auth_handler` should be freed using *GA_destroy_auth_handler*.

Note: When calling from C/C++, the parameter `details` will be emptied when the call completes.

Note: EXPERIMENTAL warning: this call may be changed in future releases.

Returns `GA_OK` or an error code.

Return type `int`

`int GA_broadcast_transaction (struct GA_session* session, const char* transaction_hex, char** tx_hash)`

Broadcast a fully signed transaction to the P2P network.

Parameters

- **session** – The session to use.
- **transaction_hex** – The signed transaction in hex to broadcast.
- **tx_hash** – Destination for the resulting transactions hash. Returned string should be freed using *GA_destroy_string*.

Returns `GA_OK` or an error code.

Return type `int`

`int GA_send_transaction (struct GA_session* session, GA_json* transaction_details, struct GA_auth_handler** call)`

Send a transaction created by `GA_create_transaction` and signed by `GA_sign_transaction`.

Parameters

- **session** – The session to use.
- **transaction_details** – The *Send transaction JSON* for sending.
- **call** – Destination for the resulting `GA_auth_handler` to perform the send. Returned `GA_auth_handler` should be freed using *GA_destroy_auth_handler*.

Note: When calling from C/C++, the parameter `transaction_details` will be emptied when the call completes.

Returns `GA_OK` or an error code.

Return type `int`

`int GA_sign_message (struct GA_session* session, GA_json* details, struct GA_auth_handler** call)`

Sign a message with the private key of an address.

Parameters

- **session** – The session to use.
- **details** – The *Sign Message JSON* detailing the message to sign and how to sign it.
- **call** – Destination for the resulting `GA_auth_handler` to perform the signing. The call handlers result is *Sign Message Result JSON*. Returned `GA_auth_handler` should be freed using *GA_destroy_auth_handler*.

Note: When calling from C/C++, the parameter `details` will be emptied when the call completes.

Returns `GA_OK` or an error code.

Return type `int`

`int GA_send_nlocktimes (struct GA_session* session)`
 Request an email containing the user's nLockTime transactions.

Parameters

- **session** – The session to use.

Returns `GA_OK` or an error code.

Return type `int`

`int GA_set_csvtime (struct GA_session* session, GA_json* locktime_details, struct GA_auth_handler** call)`
 Set the number of blocks after which CSV transactions become spendable without two factor authentication.

Parameters

- **session** – The session to use.
- **locktime_details** – The *Locktime details JSON* for setting the block value.
- **call** – Destination for the resulting `GA_auth_handler` to change the locktime. Returned `GA_auth_handler` should be freed using *GA_destroy_auth_handler*.

Note: When calling from C/C++, the parameter `locktime_details` will be emptied when the call completes.

Returns `GA_OK` or an error code.

Return type `int`

`int GA_set_nlocktime (struct GA_session* session, const GA_json* locktime_details, struct GA_auth_handler** call)`
 Set the number of blocks after which nLockTime transactions become spendable without two factor authentication. When this call succeeds, if the user has an email address associated with the wallet, an updated nlock-times.zip file will be sent via email.

Parameters

- **session** – The session to use.
- **locktime_details** – The *Locktime details JSON* for setting the block value.
- **call** – Destination for the resulting `GA_auth_handler` to change the locktime. Returned `GA_auth_handler` should be freed using *GA_destroy_auth_handler*.

Returns `GA_OK` or an error code.

Return type `int`

`int GA_set_transaction_memo (struct GA_session* session, const char* txhash_hex, const char* memo, uint32_t memo_type)`
 Add a transaction memo to a user's GreenAddress transaction.

Parameters

- **session** – The session to use.
- **txhash_hex** – The transaction hash to associate the memo with.
- **memo** – The memo to set.
- **memo_type** – Unused, pass 0.

Returns GA_OK or an error code.

Return type int

int **GA_get_fee_estimates** (struct GA_session* *session*, GA_json** *estimates*)

Get the current network's fee estimates.

Parameters

- **session** – The session to use.
- **estimates** – Destination for the returned *Fee estimates JSON*. Returned GA_json should be freed using *GA_destroy_json*.

The estimates are returned as an array of 25 elements. Each element is an integer representing the fee estimate expressed as satoshi per 1000 bytes. The first element is the minimum relay fee as returned by the network, while the remaining elements are the current estimates to use for a transaction to confirm from 1 to 24 blocks.

Returns GA_OK or an error code.

Return type int

int **GA_get_credentials** (struct GA_session* *session*, GA_json* *details*, struct GA_auth_handler** *call*)

Get the user's credentials.

Parameters

- **session** – The session to use.
- **details** – The *Get credentials JSON* to get the credentials.
- **call** – Destination for the resulting GA_auth_handler to get the user's credentials. The call handlers result is *Login credentials JSON*. Returned GA_auth_handler should be freed using *GA_destroy_auth_handler*.

Note: When calling from C/C++, the parameter *details* will be emptied when the call completes.

Returns GA_OK or an error code.

Return type int

int **GA_get_system_message** (struct GA_session* *session*, char** *message_text*)

Get the latest un-acknowledged system message.

Parameters

- **session** – The session to use.
- **message_text** – The returned UTF-8 encoded message text. Returned string should be freed using *GA_destroy_string*.

Note: If all current messages are acknowledged, an empty string is returned.

Returns GA_OK or an error code.

Return type int

int **GA_ack_system_message** (struct GA_session* session, const char* message_text, struct GA_auth_handler** call)

Sign and acknowledge a system message.

The message text will be signed with a key derived from the wallet master key and the signature sent to the server.

Parameters

- **session** – The session to use.
- **message_text** – UTF-8 encoded message text being acknowledged.
- **call** – Destination for the resulting GA_auth_handler to acknowledge the message. Returned GA_auth_handler should be freed using *GA_destroy_auth_handler*.

Returns GA_OK or an error code.

Return type int

int **GA_get_twofactor_config** (struct GA_session* session, GA_json** config)

Get the two factor configuration for the current user.

Parameters

- **session** – The session to use.
- **config** – Destination for the returned *Two-Factor config JSON*. Returned GA_json should be freed using *GA_destroy_json*.

Returns GA_OK or an error code.

Return type int

int **GA_change_settings** (struct GA_session* session, GA_json* settings, struct GA_auth_handler** call)

Change wallet settings.

Parameters

- **session** – The session to use.
- **settings** – The new *Settings JSON* values.
- **call** – Destination for the resulting GA_auth_handler. Returned GA_auth_handler should be freed using *GA_destroy_auth_handler*.

Note: When calling from C/C++, the parameter settings will be emptied when the call completes.

Returns GA_OK or an error code.

Return type int

int **GA_get_settings** (struct GA_session* session, GA_json** settings)

Get current wallet settings.

Parameters

- **session** – The session to use.

- **settings** – Destination for the current *Settings JSON*. Returned GA_json should be freed using *GA_destroy_json*.

Returns GA_OK or an error code.

Return type int

int **GA_destroy_json** (GA_json* *json*)

Free a GA_json object.

Parameters

- **json** – GA_json object to free.

Returns GA_OK or an error code.

Return type int

int **GA_auth_handler_get_status** (struct GA_auth_handler* *call*, GA_json** *output*)

Get the status/result of an action requiring authorization.

Parameters

- **call** – The auth_handler whose status is to be queried.
- **output** – Destination for the resulting *Auth handler status JSON*. Returned GA_json should be freed using *GA_destroy_json*.

Methods in the api that may require two factor or hardware authentication to complete return a GA_auth_handler object. This object encapsulates the process of determining whether authentication is required and handling conditions such as re-prompting and re-trying after an incorrect two factor code is entered.

The object acts as a state machine which is stepped through by the caller until the desired action is completed. At each step, the current state can be determined and used to perform the next action required.

Some actions require a sequence of codes and decisions; these are hidden behind the state machine interface so that callers do not need to handle special cases or program their own logic to handle any lower level API differences.

The state machine has the following states, which are returned in the "status" element from *GA_auth_handler_get_status*:

- "done": The action has been completed successfully. Any data returned from the action is present in the "result" element of the status JSON. The auth_handler object should be destroyed using *GA_destroy_auth_handler* after receiving this status.
- "error": A non-recoverable error occurred performing the action. The associated error message is given in the status element "error". The auth_handler object should be destroyed using *GA_destroy_auth_handler* and the action restarted from scratch if this state is returned.
- "request_code": Two factor authorization is required. The caller should prompt the user to choose a two factor method from the "methods" element and call *GA_auth_handler_request_code* with the selected method.
- "resolve_code": A twofactor code from the "request_code" step, or data from a hardware device is required. If the status JSON contains *Required Data JSON*, then see *GDK Hardware Wallet Interface* for details. Otherwise, to resolve a twofactor code, the caller should prompt the user to enter the code from the twofactor method chosen in the "request_code" step, and pass this code to *GA_auth_handler_resolve_code*.
- "call": Twofactor or hardware authorization is complete and the caller should call *GA_auth_handler_call* to perform the action.

Returns GA_OK or an error code.

Return type int

int **GA_auth_handler_request_code** (struct GA_auth_handler* *call*, const char* *method*)

Request a two factor authentication code to authorize an action.

Parameters

- **call** – The auth_handler representing the action to perform.
- **method** – The selected two factor method to use

Returns GA_OK or an error code.

Return type int

int **GA_auth_handler_resolve_code** (struct GA_auth_handler* *call*, const char* *code*)

Authorize an action by providing its previously requested two factor authentication code.

Parameters

- **call** – The auth_handler representing the action to perform.
- **code** – The two factor authentication code received by the user, or the serialised JSON response for hardware interaction (see [GDK Hardware Wallet Interface](#)).

Returns GA_OK or an error code.

Return type int

int **GA_auth_handler_call** (struct GA_auth_handler* *call*)

Perform an action following the completion of authorization.

Parameters

- **call** – The auth_handler representing the action to perform.

Returns GA_OK or an error code.

Return type int

int **GA_destroy_auth_handler** (struct GA_auth_handler* *call*)

Free an auth_handler after use.

Parameters

- **call** – The auth_handler to free.

Returns GA_OK or an error code.

Return type int

int **GA_change_settings_twofactor** (struct GA_session* *session*, const char* *method*, GA_json* *twofactor_details*, struct GA_auth_handler** *call*)

Enable or disable a two factor authentication method.

Parameters

- **session** – The session to use
- **method** – The two factor method to enable/disable, i.e. "email", "sms", "phone", "gauth"
- **twofactor_details** – The two factor method and associated data such as an email address. *Two-factor detail JSON*
- **call** – Destination for the resulting GA_auth_handler to perform the action Returned GA_auth_handler should be freed using [GA_destroy_auth_handler](#).

Note: When calling from C/C++, the parameter `twofactor_details` will be emptied when the call completes.

Returns `GA_OK` or an error code.

Return type `int`

`int GA_twofactor_reset (struct GA_session* session, const char* email, uint32_t is_dispute, struct GA_auth_handler** call)`

Request to begin the two factor authentication reset process.

Returns the "twofactor_reset" portion of *Two-Factor config JSON* in the `GA_auth_handler` result.

Parameters

- **session** – The session to use.
- **email** – The new email address to enable once the reset waiting period expires.
- **is_dispute** – `GA_TRUE` if the reset request is disputed, `GA_FALSE` otherwise.
- **call** – Destination for the resulting `GA_auth_handler` to request the reset. Returned `GA_auth_handler` should be freed using *GA_destroy_auth_handler*.

Returns `GA_OK` or an error code.

Return type `int`

`int GA_twofactor_undo_reset (struct GA_session* session, const char* email, struct GA_auth_handler** call)`

Undo a request to begin the two factor authentication reset process.

Returns the "twofactor_reset" portion of *Two-Factor config JSON* in the `GA_auth_handler` result.

Parameters

- **session** – The session to use.
- **email** – The email address to cancel the reset request for. Must be the email previously passed to *GA_twofactor_reset*.
- **call** – Destination for the resulting `GA_auth_handler` to request the reset. Returned `GA_auth_handler` should be freed using *GA_destroy_auth_handler*.

Note: Unlike *GA_twofactor_cancel_reset*, this call only removes the reset request associated with the given email. If other emails have requested a reset, the wallet will still remain locked following this call.

Returns `GA_OK` or an error code.

Return type `int`

`int GA_twofactor_cancel_reset (struct GA_session* session, struct GA_auth_handler** call)`

Cancel all two factor reset requests and unlock the wallet for normal operation.

This call requires authentication using an existing wallet twofactor method.

Returns the "twofactor_reset" portion of *Two-Factor config JSON* in the `GA_auth_handler` result.

Parameters

- **session** – The session to use.

- **call** – Destination for the resulting `GA_auth_handler` to cancel the reset. Returned `GA_auth_handler` should be freed using `GA_destroy_auth_handler`.

Returns `GA_OK` or an error code.

Return type `int`

`int GA_twofactor_change_limits (struct GA_session* session, GA_json* limit_details, struct GA_auth_handler** call)`

Change twofactor limits settings.

Parameters

- **session** – The session to use.
- **limit_details** – Details of the new *Transaction limits JSON*
- **call** – Destination for the resulting `GA_auth_handler` to perform the change. Returned `GA_auth_handler` should be freed using `GA_destroy_auth_handler`.

Note: When calling from C/C++, the parameter `limit_details` will be emptied when the call completes.

Returns `GA_OK` or an error code.

Return type `int`

`int GA_bcur_encode (struct GA_session* session, GA_json* details, struct GA_auth_handler** call)`

Encode CBOR into (potentially multi-part) UR-encoding.

Parameters

- **session** – The session to use.
- **details** – *BCUR Encode JSON* containing the CBOR data to encode.
- **call** – Destination for the resulting `GA_auth_handler` to complete the action. The call handlers result is *BCUR Encoded fragments JSON*. Returned `GA_auth_handler` should be freed using `GA_destroy_auth_handler`.

Note: When calling from C/C++, the parameter `details` will be emptied when the call completes.

Returns `GA_OK` or an error code.

Return type `int`

`int GA_bcur_decode (struct GA_session* session, GA_json* details, struct GA_auth_handler** call)`

Decode (potentially multi-part) UR-encoded data to CBOR.

Parameters

- **session** – The session to use.
- **details** – *BCUR Decode JSON* containing the the first URI to decode.
- **call** – Destination for the resulting `GA_auth_handler` to complete the action. The call handlers result is *BCUR Decoded data JSON*. Returned `GA_auth_handler` should be freed using `GA_destroy_auth_handler`.

For multi-part data, the call handler will request further parts using "request_code" with a method of "data". see: [Auth handler status JSON](#).

Note: When calling from C/C++, the parameter `details` will be emptied when the call completes.

Returns GA_OK or an error code.

Return type int

void **GA_destroy_string** (char* *str*)

Free a string returned by the api.

Parameters

- **str** – The string to free.

Returns GA_OK or an error code.

Return type int

int **GA_get_random_bytes** (size_t *num_bytes*, unsigned char* *output_bytes*, size_t *len*)

Get up to 32 random bytes.

Generate up to 32 random bytes using the same strategy as Bitcoin Core code.

Parameters

- **output_bytes** – bytes output buffer
- **siz** – Number of bytes to return (max. 32)

Returns GA_OK or an error code.

Return type int

int **GA_generate_mnemonic** (char** *output*)

Generate a new random BIP 39 mnemonic.

Parameters

- **output** – The generated mnemonic phrase. Returned string should be freed using [GA_destroy_string](#).

Returns GA_OK or an error code.

Return type int

int **GA_generate_mnemonic_12** (char** *output*)

Generate a new random 12 word BIP 39 mnemonic.

Parameters

- **output** – The generated mnemonic phrase. Returned string should be freed using [GA_destroy_string](#).

Returns GA_OK or an error code.

Return type int

int **GA_validate_mnemonic** (const char* *mnemonic*, uint32_t* *valid*)

Validate a BIP 39 mnemonic.

Parameters

- **mnemonic** – The mnemonic phrase

- **valid** – Destination for the result: GA_TRUE if the mnemonic is valid else GA_FALSE

Returns GA_OK or an error code.

Return type int

int **GA_register_network** (const char* *name*, const GA_json* *network_details*)

Register a network configuration

Parameters

- **name** – The name of the network to register
- **network_details** – The *Network JSON* configuration to register

Any existing configuration with the same name is overwritten. If the provided JSON is empty, any existing configuration for the network is removed.

Returns GA_OK or an error code.

Return type int

int **GA_get_networks** (GA_json** *output*)

Get the available network configurations

Parameters

- **output** – Destination for the *Network list JSON*. Returned GA_json should be freed using *GA_destroy_json*.

Returns GA_OK or an error code.

Return type int

int **GA_get_uniform_uint32_t** (uint32_t *upper_bound*, uint32_t* *output*)

Get a uint32_t in the range 0 to (upper_bound - 1) without bias

Parameters

- **output** – Destination for the generated uint32_t.

Returns GA_OK or an error code.

Return type int

This section describes the various JSON formats used by the library.

2.1 Initialization config JSON

Passed to `GA_init` when initializing the library.

```
{
  "datadir": "/path/to/store/data"
  "tordir": "/path/to/store/tor/data"
  "registrydir": "/path/to/store/registry/data"
  "log_level": "info",
}
```

datadir Mandatory. A directory which gdk will use to store encrypted data relating to sessions.

tordir An optional directory for tor state data, used when the internal tor implementation is enabled in *Connection parameters JSON*. Note that each process using the library at the same time requires its own distinct directory. If not given, a sub-directory "tor" inside "datadir" is used.

registrydir An optional directory for the registry data, used when the network is liquid based. Note that each process using the library at the same time requires its own distinct directory. If not given, a sub-directory "registry" inside "datadir" is used.

log_level Library logging level, one of "debug", "info", "warn", "error", or "none".

2.2 Connection parameters JSON

```
{
  "name": "testnet",
  "proxy": "localhost:9150",
}
```

(continues on next page)

(continued from previous page)

```
"use_tor": true,  
"user_agent": "green_android v2.33",  
"spv_enabled": false,  
"cert_expiry_threshold": 1  
}
```

name The name of the network to connect to. Must match a key from *Network list JSON*.

proxy The proxy connection to pass network traffic through, if any.

use_tor true to enable Tor connections, false otherwise. If enabled and a proxy is not given, a Tor connection will be started internally. If a proxy is given and Tor is enabled, the proxy must support resolving ".onion" domains.

user_agent The user agent string to pass to the server for multisig connections.

spv_enabled true to enable SPV verification for the session, false otherwise.

cert_expiry_threshold Ignore certificates expiring within this many days from today. Used to pre-empt problems with expiring embedded certificates.

2.3 Proxy Settings JSON

Contains the proxy settings in use by a session.

```
{  
  "proxy": "localhost:9150",  
  "use_tor": true  
}
```

proxy The proxy connection being used to pass network traffic through, or an empty string.

use_tor true if Tor is enabled, false otherwise.

2.4 Login credentials JSON

To authenticate with a hardware wallet, pass empty JSON and provide *HW device JSON*.

To authenticate with a mnemonic and optional password:

```
{  
  "mnemonic": "moral lonely ability sail balance simple kid girl inhale master_  
↪dismiss round about aerobic purpose shiver silly happy kitten track kind pattern_  
↪nose noise",  
  "password": ""  
}
```

Or, with a mnemonic and optional BIP39 passphrase:

```
{  
  "mnemonic": "moral lonely ability sail balance simple kid girl inhale master_  
↪dismiss round about aerobic purpose shiver silly happy kitten track kind pattern_  
↪nose noise",  
  "bip39_passphrase": ""  
}
```

To authenticate with a PIN:

```
{
  "pin": "123456",
  "pin_data": {
    "encrypted_data":
    ↪ "0b39c1e90ca6adce9ff35d1780de74b91d46261a7cbf2b8d2fdc21528c068c8e2b26e3bf3f6a2a992e0e1ecfad02203431",
    ↪ "pin_identifier": "38e2f188-b3a8-4d98-a7f9-6c348cb54cfe",
    "salt": "a99/9Qy6P7ON4Umk2FafVQ=="
  }
}
```

pin The PIN entered by the user to unlock the wallet.

pin_data See *PIN data JSON*.

To authenticate a watch-only user (multisig only):

```
{
  "username": "my_watch_only_username",
  "password": "my_watch_only_password"
}
```

To authenticate a watch-only wallet (singlesig and Bitcoin only):

```
{
  "core_descriptors": [ "pkh([00000000/44'/1'/0
    ↪ '])tpubDC2Q4xK4XH72J7Lkp6kAvY2Q5x4cxrKgrevkZKC2FwWZ9A9qA5eY6kvv6QDHb6iJtByzoC5J8KZZ29T45CxFz2Gh6m6PQ
    ↪ 0/*"] ,
}
```

Or alternatively:

```
{
  "slip132_extended_pubkeys": [
    ↪ "tpubDC2Q4xK4XH72J7Lkp6kAvY2Q5x4cxrKgrevkZKC2FwWZ9A9qA5eY6kvv6QDHb6iJtByzoC5J8KZZ29T45CxFz2Gh6m6PQ
    ↪ "],
}
```

The values to use for "core_descriptors" and "slip132_extended_pubkeys" can be obtained from *GA_get_subaccount*.

2.5 HW device JSON

Describes the capabilities of an external signing device.

```
{
  "device": {
    "name": "Ledger",
    "supports_ae_protocol": 0,
    "supports_arbitrary_scripts": true,
    "supports_host_unblinding": false,
    "supports_external_blinding": false,
    "supports_liquid": 1,
    "supports_low_r": false,
```

(continues on next page)

(continued from previous page)

```
}
}
```

name The unique name of the hardware device.

supports_arbitrary_scripts True if the device can sign non-standard scripts such as CSV.

supports_low_r True if the device can produce low-R ECDSA signatures.

supports_liquid 0 if the device does not support Liquid, 1 otherwise.

supports_host_unblinding True if the device supports returning the Liquid master blinding key.

supports_external_blinding True if the device supports blinding and signing Liquid transactions with outputs that are already blinded from another wallet (e.g. 2-step swaps).

supports_ae_protocol See “ae_protocol_support_level” enum in the gdk source for details.

The default for any value not provided is false or 0.

2.6 PIN data JSON

Contains the data returned by *GA_encrypt_with_pin*. The caller must persist this data and pass it to *GA_login_user* along with the users PIN in order to allow a PIN login.

```
{
  "encrypted_data":
  ↪ "0b39c1e90ca6adce9ff35d1780de74b91d46261a7cbf2b8d2fdc21528c068c8e2b26e3bf3f6a2a992e0e1ecfad0220343b",
  ↪ ",
  "pin_identifier": "38e2f188-b3a8-4d98-a7f9-6c348cb54cfe",
  "salt": "a99/9Qy6P7ON4Umk2FafVQ=="
}
```

2.7 Encrypt with PIN JSON

```
{
  "pin": "...",
  "plaintext": {}
}
```

pin The PIN to protect the server provided key.

plaintext The json to encrypt. For instance it can be the *Login credentials JSON* with the mnemonic.

2.8 Encrypt with PIN Result JSON

```
{
  "pin_data": "...",
}
```

pin_data See *PIN data JSON*.

2.9 Decrypt with PIN JSON

```
{
  "pin": "...",
  "pin_data": "...",
}
```

pin The PIN that protects the server provided key.

pin_data See *PIN data JSON*.

2.10 Wallet identifier request JSON

Describes the wallet to compute an identifier for using *GA_get_wallet_identifier*. You may pass *Login credentials JSON* to compute an identifier from a mnemonic and optional password, note that PIN or watch-only credentials cannot be used. otherwise, pass the wallets master xpub as follows:

```
{
  "master_xpub":
  ↪ "tpubD8G8MPH9RK9uk4EV97RxxhzaY8SJPuWxnViHUWji92i8B7vYdht797PPDrJveeathnKxonJe8SbaScAC1YJ8xAzZbH9Uvyv",
  ↪ " ",
}
```

master_xpub The base58-encoded BIP32 extended master public key of the wallet.

2.11 Get credentials JSON

Accepts an optional password to encrypt the mnemonic.

```
{
  "password": ""
}
```

2.12 Subaccount JSON

Describes a subaccount within the users wallet. Returned by *GA_get_subaccount* and as the array elements of *GA_get_subaccounts*.

```
{
  "hidden": false,
  "name": "Example subaccount name",
  "pointer": 0,
  "receiving_id": "GA2dlAMZojZ3TtoTsYw1JjFUeZQrft",
  "recovery_chain_code": "",
  "recovery_pub_key": "",
  "recovery_xpub": "",
  "required_ca": 0,
  "type": "2of2",
  "user_path": []
}
```

```

{
  "bip44_discovered": false,
  "core_descriptors": [
    "sh(wpkh([386e8558/49'/1'/0
↪']tpubDC2Q4xK4XH72GCMj4pLC95ZyDaset6YG6wtEpo6avjWg3uP25RskxXYt9WeNbCEWrd183N6hb5dGhsrs9y8f9JFXx3V
↪0/*))#f15guxrs",
    "sh(wpkh([386e8558/49'/1'/0
↪']tpubDC2Q4xK4XH72GCMj4pLC95ZyDaset6YG6wtEpo6avjWg3uP25RskxXYt9WeNbCEWrd183N6hb5dGhsrs9y8f9JFXx3V
↪1/*))#u767yek0"
  ],
  "hidden": false,
  "name": "Example subaccount name",
  "pointer": 0,
  "receiving_id": "",
  "required_ca": 0,
  "slip132_extended_pubkey":
↪"upub5D9ydiUdMxX8SWbX7YzEJtx6NZLf4r4hoyjH84MnSod99KGdFEXywtudqrgq4yKWqo7oBMo9fzjaMqXa8CsrNzo9sLCut
↪",
  "type": "p2sh-p2wpkh",
  "user_path": [
    2147483697,
    2147483649,
    2147483648
  ]
}

```

```

{
  "hidden": false,
  "name": "Example subaccount name",
  "pointer": 0,
  "receiving_id": "GA2G8fjDNWvZiJP8Y5iV2ywTAQeZeo",
  "recovery_chain_code": "",
  "recovery_pub_key": "",
  "recovery_xpub": "",
  "required_ca": 0,
  "type": "2of2",
  "user_path": []
}

```

```

{
  "bip44_discovered": false,
  "core_descriptors": [
    "sh(wpkh([386e8558/49'/1'/0
↪']tpubDC2Q4xK4XH72GCMj4pLC95ZyDaset6YG6wtEpo6avjWg3uP25RskxXYt9WeNbCEWrd183N6hb5dGhsrs9y8f9JFXx3V
↪0/*))#f15guxrs",
    "sh(wpkh([386e8558/49'/1'/0
↪']tpubDC2Q4xK4XH72GCMj4pLC95ZyDaset6YG6wtEpo6avjWg3uP25RskxXYt9WeNbCEWrd183N6hb5dGhsrs9y8f9JFXx3V
↪1/*))#u767yek0"
  ],
  "hidden": false,
  "name": "Example subaccount name",
  "pointer": 0,
  "receiving_id": "",
  "required_ca": 0,
  "type": "p2sh-p2wpkh",
  "user_path": [
    2147483697,

```

(continues on next page)

(continued from previous page)

```

2147483649,
2147483648
]
}

```

hidden Whether the subaccount is hidden.

name The name of the subaccount.

pointer The subaccount number.

receiving_id The Green receiving ID for the subaccount.

recovery_chain_code For "2of3" subaccounts, the BIP32 chain code of the users recovery key.

recovery_pub_key For "2of3" subaccounts, the BIP32 public key of the users recovery key.

recovery_xpub For "2of3" subaccounts, the BIP32 xpub of the users recovery key.

required_ca For "2of2_no_recovery" subaccounts, the number of confidential addresses that the user must upload to the server before transacting.

type For multisig subaccounts, one of "2of2", "2of3" or "2of2_no_recovery". For singlesig subaccounts, one of "p2pkh", "p2wpkh" or "p2sh-p2wpkh".

bip44_discovered Singlesig only. Whether or not this subaccount contains at least one transaction.

user_path The BIP32 path for this subaccount. This field is only returned by [GA_get_subaccount](#).

core_descriptors Singlesig only. The Bitcoin Core compatible output descriptors.

One for the external chain and one for internal chain (change), for instance

"sh(wpkh(tpubDC2Q4xK4XH72H18SiEV2A6HUwUPLhXiTEQXU35r4a41ZVrUv2cgKUMm2fsKTapi8DH4Y820/*))" "sh(wpkh(tpubDC2Q4xK4XH72H18SiEV2A6HUwUPLhXiTEQXU35r4a41ZVrUv2cgKUMm2fsKTapi1/*))" for a p2sh-p2wpkh subaccount. This field is only returned by [GA_get_subaccount](#).

slip132_extended_pubkey Singlesig and Bitcoin only. The extended public key with modified version as specified in SLIP-0132 (xpub, ypub, zpub, tpub, upub, vpub). Use of this value is discouraged and this field might be removed in the future. Callers should use descriptors instead. This field is only returned by [GA_get_subaccount](#).

2.13 Subaccount update JSON

Describes updates to be made to a subaccount via [GA_update_subaccount](#).

```

{
  "hidden": true,
  "name": "New name",
  "subaccount": 1
}

```

hidden If present, updates whether the subaccount will be marked hidden.

name If present, updates the name of the subaccount.

subaccount The subaccount to update.

2.14 Subaccounts list JSON

```
{
  "subaccounts": [
    { },
    { }
  ]
}
```

subaccounts An array of *Subaccount JSON* elements for each of the users subaccounts.

2.15 Transaction list JSON

Describes a users transaction history returned by *GA_get_transactions*.

```
{
  "transactions": [
    {
      "block_height": 0,
      "can_cpfp": true,
      "can_rbf": false,
      "created_at_ts": 1686640441208984,
      "fee": 1430,
      "fee_rate": 10000,
      "inputs": [
        {
          "address": "bort1qsjtt6hv699fulumf0ravymecm8792shd6hz4kz",
          "address_type": "",
          "is_internal": false,
          "is_output": false,
          "is_relevant": false,
          "is_spent": true,
          "pointer": 0,
          "pt_idx": 0,
          "satoshi": 5000000000,
          "script_type": 160,
          "subaccount": 0,
          "subtype": 0
        }
      ],
      "memo": "",
      "outputs": [
        {
          "address": "2N2zyKJEcAtqgHRdYFqCQDKZozDi5GY9wi5",
          "address_type": "csv",
          "is_internal": false,
          "is_output": true,
          "is_relevant": true,
          "is_spent": false,
          "pointer": 1,
          "pt_idx": 0,
          "satoshi": 12340,
          "script_type": 15,
          "subaccount": 0,
          "subtype": 51840
        }
      ]
    }
  ]
}
```

(continues on next page)

(continued from previous page)

```

    },
    {
      "address": "2MuhZUoJcSa2XwjejYmrBkA2x5WqM7UcZgg",
      "address_type": "",
      "is_internal": false,
      "is_output": true,
      "is_relevant": false,
      "is_spent": false,
      "pointer": 0,
      "pt_idx": 1,
      "satoshi": 4999986230,
      "script_type": 9,
      "subaccount": 0,
      "subtype": 0
    }
  ],
  "rbf_optin": false,
  "satoshi": {
    "btc": 12340
  },
  "spv_verified": "disabled",
  "transaction_vsize": 143,
  "transaction_weight": 572,
  "txhash": "1a45e3a1140b17b45277e7e1189ff8385c7e0fd4cd88a5840b15ee600d5dbc30",
  "type": "incoming"
}
]
}

```

```

{
  "transactions": [
    {
      "block_height": 0,
      "can_cpfp": false,
      "can_rbf": false,
      "created_at_ts": 1686597244463054,
      "fee": 1430,
      "fee_rate": 10052,
      "inputs": [
        {
          "address": "bcr11qsjtt6hv699fulumf0ravymecm8792shd6hz4kz",
          "address_type": "",
          "is_internal": false,
          "is_output": false,
          "is_relevant": false,
          "is_spent": true,
          "pointer": 0,
          "pt_idx": 0,
          "satoshi": 5000000000,
          "script": "00148496bd5d9a2953cff36978fac26f38d9fc5542ed",
          "script_type": 0,
          "subaccount": 0,
          "subtype": 0
        }
      ],
      "memo": "",

```

(continues on next page)

(continued from previous page)

```

"outputs": [
  {
    "address": "2N2whnPNT4fYtntAYNTWoCshDVBphCRzyH7",
    "address_type": "",
    "is_internal": false,
    "is_output": true,
    "is_relevant": false,
    "is_spent": false,
    "pointer": 0,
    "pt_idx": 0,
    "satoshi": 4999986230,
    "script": "a9146a60f8f6cd77c44582c04b8912dcba06601503e087",
    "script_type": 0,
    "subaccount": 0,
    "subtype": 0
  },
  {
    "address": "2MsHqxhz5fHt2QyvE8TgT1MvzDY9kdMH5Rz",
    "address_type": "p2sh-p2wpkh",
    "is_internal": false,
    "is_output": true,
    "is_relevant": true,
    "is_spent": false,
    "pointer": 1,
    "pt_idx": 1,
    "satoshi": 12340,
    "script": "a914007edc4b1153a320e490b5fe29a33e95af2d3c7387",
    "script_type": 0,
    "subaccount": 0,
    "subtype": 0
  }
],
"rbf_optin": false,
"satoshi": {
  "btc": 12340
},
"spv_verified": "disabled",
"transaction_vsize": 143,
"transaction_weight": 569,
"txhash": "8a3974692b29237cb81f545a7891be89d73aed7d0038879da437623b0fd57584",
"type": "incoming"
}
]
}

```

```

{
  "transactions": [
    {
      "block_height": 0,
      "can_cpfp": true,
      "can_rbf": false,
      "created_at_ts": 1686642370218278,
      "fee": 24920,
      "fee_rate": 10000,
      "inputs": [
        {

```

(continues on next page)

(continued from previous page)

```

    "address": "ert1qer759naur22vw6tnssc2ey3eqg974um62nh0cq",
    "address_type": "",
    "is_internal": false,
    "is_output": false,
    "is_relevant": false,
    "is_spent": true,
    "pointer": 0,
    "pt_idx": 0,
    "script_type": 160,
    "subaccount": 0,
    "subtype": 0
  }
],
"memo": "",
"outputs": [
  {
    "address":
    ↪ "AzpsXif2CBq9WETyZW5JR85LUEBXjvZXi2wx5EaKqXQ1cQh8Jo35fGva7MWcfQmkNZ3D2hZbXD3U16eF",
    "address_type": "csv",
    "amountblinder":
    ↪ "337fbb7a7c554a1a11afd58524a650811e6a2e655966e5e8870c6aa5bc220c3",
    "asset_id":
    ↪ "5ac9f65c0efcc4775e0baec4ec03abdde22473cd3cf33c0419ca290e0751b225",
    "asset_tag":
    ↪ "0aec3e557312bbe41d1ead85eb9e2a741965f5fb7426a398c882c8bb9b6eb0a1d9",
    "assetblinder":
    ↪ "0ea4abfc0c7c36929a6d051d9694ac58a5c57a571af89bf34337a02e29552b7a",
    "blinding_key":
    ↪ "033c13b87298d77941d6e4ec5c754a03a41a7c48ad4196e1e97a734d8646daa231",
    "commitment":
    ↪ "09f9d304ce60ad52731ea60b64e069623c4df719062cc49f8d3a9e3ca06ef411b2",
    "is_blinded": true,
    "is_confidential": true,
    "is_internal": false,
    "is_output": true,
    "is_relevant": true,
    "is_spent": false,
    "nonce_commitment":
    ↪ "03b4ff371f4545ba5f1f9b58c6fea3ea99939c86f206ca594277a8d24d2d7333ed",
    "pointer": 1,
    "pt_idx": 0,
    "satoshi": 12340,
    "script": "a91469a22b9f6314c0fe6db92286592ec287ac52c03587",
    "script_type": 15,
    "subaccount": 0,
    "subtype": 65535,
    "unconfidential_address": "XLyn55q1TtektRH3YrDTT8vMTMkjrcT38u"
  },
  {
    "address": "ert1q0exrx3qfddfjgsv3un8vq0wkh7k4mvjyq48mcc",
    "address_type": "",
    "asset_tag":
    ↪ "0a633e984d81a08b8031a688ea5bea540fcdf08b0fb1a2c6ed1242b934804a40ab",
    "commitment":
    ↪ "085aa714e5253c74d5701770d5c2e97f039c4780036c53dbb0b074659edfe18489",
    "is_internal": false,
    "is_output": true,

```

(continues on next page)

(continued from previous page)

```

        "is_relevant": false,
        "is_spent": false,
        "nonce_commitment":
↪ "025f28d0f5766008f1d2bbd19fcd54851d6c754bcef04ba6879649250f281fef2c",
        "pointer": 0,
        "pt_idx": 1,
        "script": "00147e4c3344096b53244191e4cec03dd6bfad5db244",
        "script_type": 11,
        "subaccount": 0,
        "subtype": 0
    },
    {
        "address": "",
        "address_type": "",
        "asset_tag":
↪ "0125b251070e29ca19043cf33ccd7324e2ddab03ecc4ae0b5e77c4fc0e5cf6c95a",
        "commitment": "010000000000006158",
        "is_internal": false,
        "is_output": true,
        "is_relevant": false,
        "is_spent": false,
        "nonce_commitment":
↪ "0000000000000000000000000000000000000000000000000000000000000000",
        "pointer": 0,
        "pt_idx": 2,
        "script": "",
        "script_type": 0,
        "subaccount": 0,
        "subtype": 0
    }
],
"rbf_optin": true,
"satoshi": {
    "5ac9f65c0efcc4775e0baec4ec03abdde22473cd3cf33c0419ca290e0751b225": 12340
},
"spv_verified": "disabled",
"transaction_vsize": 2492,
"transaction_weight": 9968,
"txhash": "46dd2e84591e3a27c85d3ead917a1469dd09458fb1d76bce6b58885bf5d6ebbc",
"type": "incoming"
}
]
}

```

```

{
  "transactions": [
    {
      "block_height": 0,
      "can_cpfp": false,
      "can_rbf": false,
      "created_at_ts": 1686599171553911,
      "fee": 24920,
      "fee_rate": 10002,
      "inputs": [
        {
          "address": "",

```

(continues on next page)

(continued from previous page)

```

    "address_type": "",
    "is_internal": false,
    "is_output": false,
    "is_relevant": false,
    "is_spent": true,
    "pointer": 0,
    "pt_idx": 0,
    "satoshi": 0,
    "script": "",
    "script_type": 0,
    "subaccount": 0,
    "subtype": 0
  }
],
"memo": "",
"outputs": [
  {
    "address": "ert1q82q9y0a1l6c08axd67yw9d85pea05szhh6jgzj",
    "address_type": "",
    "is_internal": false,
    "is_output": true,
    "is_relevant": false,
    "is_spent": false,
    "pointer": 0,
    "pt_idx": 0,
    "satoshi": 0,
    "script": "00143a80523fbffeb0f3f4cdd788e2b4f40e7afa4057",
    "script_type": 0,
    "subaccount": 0,
    "subtype": 0
  },
  {
    "address":
    ↪ "AzpjM6F3vhwPK3U5p8u79osjhbhsduwdGCrbt1Cahy8nD3d7SbQJte99KiMkxWZLBrmUnWPrU3eLtUbVU",
    "address_type": "p2sh-p2wpkh",
    "amountblinder":
    ↪ "8efb0d0b21e6f14960f4c2affc166e50555f03bfc5ece3f2992b6ab4fd975caf",
    "asset_id":
    ↪ "5ac9f65c0efcc4775e0baec4ec03abdde22473cd3cf33c0419ca290e0751b225",
    "assetblinder":
    ↪ "a986f588d8f05e337627bc780732efde594ad3cf32cb17e994dc7dcb25bc72fd",
    "blinding_key":
    ↪ "020de79e60d9d17b22c9f803455eb2fc48becc50fe76d32508bde357bf10c28a8a",
    "is_blinded": true,
    "is_confidential": true,
    "is_internal": false,
    "is_output": true,
    "is_relevant": true,
    "is_spent": false,
    "pointer": 1,
    "pt_idx": 1,
    "satoshi": 12340,
    "script": "a914007edc4b1153a320e490b5fe29a33e95af2d3c7387",
    "script_type": 0,
    "subaccount": 0,
    "subtype": 0,
    "unconfidential_address": "XBPrpZvDkTp1TY3kAhNtLBzo4Vycb9BsQt"
  }
]

```

(continues on next page)

(continued from previous page)

```

    },
    {
      "address": "",
      "address_type": "",
      "is_internal": false,
      "is_output": true,
      "is_relevant": false,
      "is_spent": false,
      "pointer": 0,
      "pt_idx": 2,
      "satoshi": 0,
      "script": "",
      "script_type": 0,
      "subaccount": 0,
      "subtype": 0
    }
  ],
  "rbf_optin": true,
  "satoshi": {
    "5ac9f65c0efcc4775e0baec4ec03abdde22473cd3cf33c0419ca290e0751b225": 12340
  },
  "spv_verified": "disabled",
  "transaction_vsize": 2492,
  "transaction_weight": 9966,
  "txhash": "eea0ccc14f823dfcf6afcaf9f57db43aac294d675dd19bb3b563fc11ee169864",
  "type": "incoming"
}
]
}

```

transactions Top level container for the users transaction list.

block_height The network block height that the transaction was confirmed in, or 0 if the transaction is in the mempool.

can_cpfp A boolean indicating whether the user can CPFP the transaction.

can_rbf A boolean indicating whether the use can RBF (bump) the transaction fee.

created_at_ts The timestamp in microseconds from the Unix epoch when the transaction was seen by gdk or Green servers, or included in a block.

fee The BTC or L-BTC network fee paid by the transaction in satoshi.

fee_rate The fee rate in satoshi per thousand bytes.

inputs See *Transaction list input element*.

memo The users memo, if previously set by *GA_set_transaction_memo*.

outputs See *Transaction list output element*.

rbf_optin A boolean indicating whether the transaction is RBF-enabled.

satoshi A map of asset names to the signed satoshi total for that asset in the transaction. Negative numbers represent outgoing amounts, positive incoming.

spv_verified The SPV status of the transaction, one of "in_progress", "verified", "not_verified", "disabled", "not_longest" or "unconfirmed".

transaction_vsize The size of the transaction in vbytes.

transaction_weight The weight of the transaction.

txhash The txid of the transaction.

type One of "incoming", "outgoing", "mixed" or "not unblindable".

2.16 Transaction list input element

Describes a transaction input in *Transaction list JSON*.

```
{
  "address": "bcrt1qsjtt6hv699fulumf0ravymecm8792shd6hz4kz",
  "address_type": "",
  "is_internal": false,
  "is_output": false,
  "is_relevant": false,
  "is_spent": true,
  "pointer": 0,
  "pt_idx": 0,
  "satoshi": 5000000000,
  "script_type": 160,
  "subaccount": 0,
  "subtype": 0
}
```

```
{
  "address": "bcrt1qsjtt6hv699fulumf0ravymecm8792shd6hz4kz",
  "address_type": "",
  "is_internal": false,
  "is_output": false,
  "is_relevant": false,
  "is_spent": true,
  "pointer": 0,
  "pt_idx": 0,
  "satoshi": 5000000000,
  "script": "00148496bd5d9a2953cff36978fac26f38d9fc5542ed",
  "script_type": 0,
  "subaccount": 0,
  "subtype": 0
}
```

```
{
  "address": "ert1qer759naur22vw6tnssc2ey3eqg974um62nh0cq",
  "address_type": "",
  "is_internal": false,
  "is_output": false,
  "is_relevant": false,
  "is_spent": true,
  "pointer": 0,
  "pt_idx": 0,
  "script_type": 160,
  "subaccount": 0,
  "subtype": 0
}
```

```
{
  "address": "",
  "address_type": "",
  "is_internal": false,
  "is_output": false,
  "is_relevant": false,
  "is_spent": true,
  "pointer": 0,
  "pt_idx": 0,
  "satoshi": 0,
  "script": "",
  "script_type": 0,
  "subaccount": 0,
  "subtype": 0
}
```

address For user wallet addresses, the wallet address in base58, bech32 or blech32 encoding.

addressee Optional, multisig only. For historical social payments, the account name sent from.

address_type For user wallet addresses, One of "csv", "p2sh", "p2wsh" (multisig), or "p2pkh", "p2sh-p2wpkh", "p2wpkh" (singlesig), indicating the type of address.

is_internal Whether or not the user key belongs to the internal chain. Always false for multisig.

is_output Always false. Deprecated, will be removed in a future release.

is_relevant A boolean indicating whether the input relates to the subaccount the caller passed to *GA_get_transactions*.

is_spent Always true. Deprecated, will be removed in a future release.

pointer For user wallet addresses, the address number/final number in the address derivation path.

pt_idx Deprecated, will be removed in a future release.

satoshi The amount of the input in satoshi.

script_type Deprecated, will be removed in a future release.

subaccount For user wallet addresses, the subaccount this output belongs to, or 0.

subtype For "address_type" "csv", the number of CSV blocks used in the receiving scriptpubkey.

Liquid inputs have additional fields:

amountblinder The hex-encoded amount blinder (value blinding factor, vbf).

asset_id The hex-encoded asset id in display format.

asset_tag The hex-encoded asset commitment.

assetblinder The hex-encoded asset blinder (asset blinding factor, abf).

commitment The hex-encoded value commitment.

is_blinded A boolean indicating whether or not the input is blinded.

nonce_commitment The hex-encoded nonce commitment.

previdx The output index of the transaction containing the output representing this input.

prevpointer Deprecated, will be removed in a future release.

prevsubaccount Deprecated, will be removed in a future release.

prevtxhash The txid of the transaction containing the output representing this input.

script The scriptpubkey of the output representing this input.

2.17 Transaction list output element

Describes a transaction output in *Transaction list JSON*.

```
{
  "address": "2N2zyKJEcAtqgHRdYFqCQDKZozDi5GY9wi5",
  "address_type": "csv",
  "is_internal": false,
  "is_output": true,
  "is_relevant": true,
  "is_spent": false,
  "pointer": 1,
  "pt_idx": 0,
  "satoshi": 12340,
  "script_type": 15,
  "subaccount": 0,
  "subtype": 51840
}
```

```
{
  "address": "2MsHqxhz5fHt2QyvE8TgT1MvzDY9kdMH5Rz",
  "address_type": "p2sh-p2wpkh",
  "is_internal": false,
  "is_output": true,
  "is_relevant": true,
  "is_spent": false,
  "pointer": 1,
  "pt_idx": 1,
  "satoshi": 12340,
  "script": "a914007edc4b1153a320e490b5fe29a33e95af2d3c7387",
  "script_type": 0,
  "subaccount": 0,
  "subtype": 0
}
```

```
{
  "address":
  ↪ "AzpsXif2CBq9WEtyZW5JR85LUEBXjvZXi2wx5EaKqXQ1cQh8Jo35fGva7MWcfQmkNZ3D2hZbXD3U16eF",
  "address_type": "csv",
  "amountblinder": "337fbb7a7c554a1a11a1fd58524a650811e6a2e655966e5e8870c6aa5bc220c3",
  "asset_id": "5ac9f65c0efcc4775e0baec4ec03abdde22473cd3cf33c0419ca290e0751b225",
  "asset_tag": "0aec3e557312bbe41dlead85eb9e2a741965f5fb7426a398c882c8bb9b6eb0a1d9",
  "assetblinder": "0ea4abfc0c7c36929a6d051d9694ac58a5c57a571af89bf34337a02e29552b7a",
  "blinding_key": "033c13b87298d77941d6e4ec5c754a03a41a7c48ad4196e1e97a734d8646daa231
  ↪ ",
  "commitment": "09f9d304ce60ad52731ea60b64e069623c4df719062cc49f8d3a9e3ca06ef411b2",
  "is_blinded": true,
  "is_confidential": true,
  "is_internal": false,
  "is_output": true,
  "is_relevant": true,
  "is_spent": false,
  "nonce_commitment":
  ↪ "03b4ff371f4545ba5f1f9b58c6fea3ea99939c86f206ca594277a8d24d2d7333ed",
  (continues on next page)
```

(continued from previous page)

```

"pointer": 1,
"pt_idx": 0,
"satoshi": 12340,
"script": "a91469a22b9f6314c0fe6db92286592ec287ac52c03587",
"script_type": 15,
"subaccount": 0,
"subtype": 65535,
"unconfidential_address": "XLyn55q1TtektRH3YrDTT8vMTMkjrcT38u"
}

{
  "address":
  ↪ "Azpjm6F3vhwPK3U5p8u79osjbhsduwdGCrbt1Cahy8nD3d7SbQJte99KiMkxWZLBrmUnWPrU3eLtUbVU",
  "address_type": "p2sh-p2wpkh",
  "amountblinder": "8efb0d0b21e6f14960f4c2affc166e50555f03bfc5ece3f2992b6ab4fd975caf",
  "asset_id": "5ac9f65c0efcc4775e0baec4ec03abdde22473cd3cf33c0419ca290e0751b225",
  "assetblinder": "a986f588d8f05e337627bc780732efde594ad3cf32cb17e994dc7dcb25bc72fd",
  "blinding_key": "020de79e60d9d17b22c9f803455eb2fc48becc50fe76d32508bde357bf10c28a8a
  ↪ ",
  "is_blinded": true,
  "is_confidential": true,
  "is_internal": false,
  "is_output": true,
  "is_relevant": true,
  "is_spent": false,
  "pointer": 1,
  "pt_idx": 1,
  "satoshi": 12340,
  "script": "a914007edc4b1153a320e490b5fe29a33e95af2d3c7387",
  "script_type": 0,
  "subaccount": 0,
  "subtype": 0,
  "unconfidential_address": "XBPrpZvDkTp1TY3kAhNtLBzo4Vycb9BsQt"
}

```

address For user wallet addresses, the wallet address in base58, bech32 or blech32 encoding.

address_type For user wallet output addresses, One of "csv", "p2sh", "p2wsh" (multisig), or "p2pkh", "p2sh-p2wpkh", "p2wpkh" (singlesig), indicating the type of address.

is_internal Whether or not the user key belongs to the internal chain. Always false for multisig.

is_output Always true. Deprecated, will be removed in a future release.

is_relevant A boolean indicating whether the output relates to the subaccount the caller passed to *GA_get_transactions*.

is_spent A boolean indicating if this output has been spent.

pointer For user wallet addresses, the address number/final number in the address derivation path.

pt_idx Deprecated, will be removed in a future release.

satoshi The amount of the output in satoshi.

script_type Deprecated, will be removed in a future release.

subaccount For user wallet addresses, the subaccount this output belongs to, or 0.

subtype For "address_type" "csv", the number of CSV blocks used in the receiving scriptpubkey.

Liquid outputs have the following additional fields:

- amountblinder** The hex-encoded amount blinder (value blinding factor, vbf).
- asset_id** The hex-encoded asset id in display format.
- asset_tag** The hex-encoded asset commitment.
- assetblinder** The hex-encoded asset blinder (asset blinding factor, abf).
- blinding_key** The blinding public key for the output.
- commitment** The hex-encoded value commitment.
- is_blinded** For user wallet outputs, a boolean indicating whether or not the output is blinded.
- is_confidential** For user wallet outputs, always true when `is_blinded` is true.
- nonce_commitment** The hex-encoded nonce commitment.
- script** For user wallet outputs, the scriptpubkey of this output.
- unconfidential_address** For user wallet outputs, the non-confidential address corresponding to address. This is provided for informational purposes only and should not be used to receive.

2.18 Transaction details JSON

Contains information about a transaction that may not be associated with the users wallet. Returned by `GA_get_transaction_details`.

```
{
  "transaction":
  ↪ "0200000000001019b763a75636ba2deaefc5cc5cbb7dc201f45de5d2aa1ffcda6636ecc4528f5ca000000000feffffff02",
  ↪ "transaction_locktime": 200,
  "transaction_version": 2,
  "transaction_vsize": 143,
  "transaction_weight": 569,
  "txhash": "1a45e3a1140b17b45277e7e1189ff8385c7e0fd4cd88a5840b15ee600d5dbc30"
}
```

```
{
  "transaction":
  ↪ "020000000000101c7fc89a3a6b1568916f64c9e9abd77bd53d16daa4de758af8d8a54d19c45b995000000000feffffff02",
  ↪ "transaction_locktime": 200,
  "transaction_version": 2,
  "transaction_vsize": 143,
  "transaction_weight": 569,
  "txhash": "8a3974692b29237cb81f545a7891be89d73aed7d0038879da437623b0fd57584"
}
```

```
{
  "transaction": "Transaction Hex, abbreviated here for length",
  "transaction_locktime": 200,
  "transaction_version": 2,
  "transaction_vsize": 2492,
  "transaction_weight": 9966,
}
```

(continues on next page)

(continued from previous page)

```
}
  "txhash": "46dd2e84591e3a27c85d3ead917a1469dd09458fb1d76bce6b58885bf5d6ebbc"
}
```

```
{
  "transaction": "Transaction Hex, abbreviated here for length",
  "transaction_locktime": 200,
  "transaction_version": 2,
  "transaction_vsize": 2492,
  "transaction_weight": 9966,
  "txhash": "eea0ccc14f823dfcf6afcaf9f57db43aac294d675dd19bb3b563fc11ee169864"
}
```

2.19 Sign transaction JSON

Contains the details of a caller-generated transaction to sign.

To sign with a specific sighash, set "user_sighash" for the elements of "transaction_inputs" you wish to sign with a certain sighash, otherwise SIGHASH_ALL (1) will be used.

Set "skip_signing" to true for any input in "transaction_inputs" you do not wish to have signed.

All other fields are not user-editable and should be passed unchanged.

2.20 Send transaction JSON

Contains the details of a caller-generated and signed transaction from *GA_sign_transaction* to send to the network.

For multisig session, this will send via the Green backend service, signing any inputs that require service signatures before broadcasting.

All fields are not user-editable and should be passed unchanged.

2.21 Create Swap Transaction JSON

Describes the swap to be created when calling *GA_create_swap_transaction*.

```
{
  "swap_type": "liquidex",
  "input_type": "liquidex_v1",
  "liquidex_v1": {},
  "output_type": "liquidex_v1"
}
```

swap_type Pass "liquidex" to create the maker's side of a LiquidEX 2-step swap.

input_type Pass "liquidex_v1" to pass LiquidEX version 1 details.

liquidex_v1 The LiquidEX v1 specific parameters, see *LiquidEX Create Swap transaction JSON*. This field must included only if "input_type" is "liquidex_v1".

output_type Pass "liquidex_v1" to return LiquidEX proposal JSON version 1.

2.22 Create Swap Transaction Result JSON

If the "output_type" was "liquidex_v1" this field is *LiquidEX Create Swap Transaction Result JSON*.

2.23 Complete Swap Transaction JSON

Describes the swap to be completed when calling *GA_complete_swap_transaction*.

```
{
  "swap_type": "liquidex",
  "input_type": "liquidex_v1",
  "liquidex_v1": {},
  "output_type": "transaction",
  "utxos": {},
}
```

swap_type Pass "liquidex" to complete the taker's side of a LiquidEX 2-step swap.

input_type Pass "liquidex_v1" to pass a LiquidEX proposal JSON version 1.

liquidex_v1 The LiquidEX v1 specific parameters, see *LiquidEX Complete Swap transaction JSON*. This field must included only if "input_type" is "liquidex_v1".

output_type Pass "transaction" to return a transaction JSON that can be passed to *GA_sign_transaction*.

utxos Mandatory. The UTXOs to fund the transaction with, *Unspent outputs JSON* as returned by *GA_get_unspent_outputs*. Note that coin selection is not performed on the passed UTXOs. All passed UTXOs of the same asset as the receiving asset id will be included in the transaction.

2.24 Complete Swap Transaction Result JSON

If the "output_type" was "transaction" this field is *Sign transaction JSON*.

2.25 Sign PSBT JSON

```
{
  "psbt": "...",
  "utxos": [],
  "blinding_nonces": [],
}
```

psbt The PSBT or PSET encoded in base64 format.

utxos Mandatory. The UTXOs that should be signed, *Unspent outputs JSON* as returned by *GA_get_unspent_outputs*. UTXOs that are not inputs of the PSBT/PSET can be included. Caller can avoid signing an input by not passing in its UTXO.

blinding_nonces For "2of2_no_recovery" subaccounts only, the blinding nonces in hex format for all outputs.

2.26 Sign PSBT Result JSON

```
{
  "psbt": "...",
  "utxos": [],
}
```

psbt The input PSBT or PSET in base64 format, with signatures added for all inputs signed.

utxos The UTXOs corresponding to each signed input, in the order they appear in the PSBT transaction.

2.27 PSBT Get Details JSON

```
{
  "psbt": "...",
  "utxos": [],
}
```

psbt The PSBT or PSET encoded in base64 format.

utxos Mandatory. The UTXOs owned by the wallet, *Unspent outputs JSON* as returned by *GA_get_unspent_outputs*. UTXOs that are not inputs of the PSBT/PSET can be included.

2.28 PSBT Get Details Result JSON

```
{
  "inputs": [
    {
      "asset_id": "...",
      "satoshi": 0,
      "subaccount": 0,
    },
  ],
  "outputs": [
    {
      "asset_id": "...",
      "satoshi": 0,
      "subaccount": 0,
    },
  ],
}
```

Note: Inputs and outputs might have additional fields that might be removed or changed in following releases.

2.29 Sign Message JSON

Describes a request for the wallet to sign a given message via *GA_sign_message*.

```
{
  "address": "...",
  "message": "...",
}
```

address The address to use for the private key. Must be a P2PKH address, and the address must belong to the wallet.

message The message to sign.

2.30 Sign Message Result JSON

Returned by *GA_sign_message*.

```
{
  "signature": "...",
}
```

signature The recoverable signature of the message encoded in base 64.

2.31 Fee estimates JSON

```
{ "fees": [1000,10070,10070,10070,3014,3014,3014,2543,2543,2543,2543,2543,2543,1499,
→1499,1499,1499,1499,1499,1499,1499,1499,1499,1499,1499] }
```

2.32 Two-Factor config JSON

Describes the wallets enabled two factor methods, current spending limits, and two factor reset status.

```
{
  "all_methods": [
    "email",
    "sms",
    "phone",
    "gauth"
  ],
  "any_enabled": true,
  "email": {
    "confirmed": true,
    "data": "***@@g***",
    "enabled": true
  },
  "enabled_methods": [
    "email"
  ],
  "gauth": {
    "confirmed": false,
    "data": "otpaath://totp/Green%20Bitcoin?secret=IZ3SMET5RDWVUSHB4CPTKUWBJM4HSYHO",
    "enabled": false
  },
}
```

(continues on next page)

(continued from previous page)

```

"limits": {
  "bits": "5000.00",
  "btc": "0.00500000",
  "fiat": "0.01",
  "fiat_currency": "EUR",
  "fiat_rate": "1.10000000",
  "is_fiat": false,
  "mbtc": "5.00000",
  "satoshi": 500000,
  "sats": "500000",
  "ubtc": "5000.00"
},
"phone": {
  "confirmed": false,
  "data": "",
  "enabled": false
},
"sms": {
  "confirmed": false,
  "data": "",
  "enabled": false
},
"twofactor_reset": {
  "days_remaining": -1,
  "is_active": false,
  "is_disputed": false
}
}

```

When the user has a fiat spending limit set instead of BTC, limits are returned as e.g:

```

"limits": {
  "fiat": "0.01",
  "fiat_currency": "EUR",
  "is_fiat": true
}

```

twofactor_reset/days_remaining The number of days remaining before the wallets two factor authentication is reset, or -1 if no reset procedure is underway.

twofactor_reset/is_active Whether or not the wallet is currently undergoing the two factor reset procedure.

twofactor_reset/is_disputed Whether or not the wallet two factor reset procedure is disputed.

2.33 BCUR Encode JSON

Contains CBOR data to encode into UR format using *GA_bcur_encode*.

```

{
  "ur_type": "crypto-seed",
  "data": "A20150C7098580125E2AB0981253468B2DBC5202D8641947DA",
  "max_fragment_len": 100
}

```

ur_type The type of the CBOR-encoded data.

data CBOR-encoded data in hexadecimal format.

max_fragment_len The maximum size of each UR-encoded fragment to return.

Where `data` is longer than `max_fragment_len`, the result is a multi-part encoding using approximately 3 times the minimum number of fragments needed to decode the data, split into parts of size `max_fragment_len` or less.

In this case, the caller must provide all returned parts to any decoder, e.g. by generating an animated QR code from them.

2.34 BCUR Encoded fragments JSON

Contains UR format data encoded using `GA_bcur_encode`.

```
{
  "parts": ["ur:crypto-seed/
↪oadgdstaslpabghydrpfmkbggufgludprfgmaotpiecffltlnpqdenos"]
}
```

parts The resulting array of UR-encoded fragments representing the input CBOR.

2.35 BCUR Decode JSON

Contains UR encoded data to decode into CBOR using `GA_bcur_decode`.

```
{
  "part": "ur:crypto-seed/oadgdstaslpabghydrpfmkbggufgludprfgmaotpiecffltlnpqdenos"
}
```

part The UR-encoded string for an individual part. For multi-part decoding, the parts can be provided in any order.

2.36 BCUR Decoded data JSON

Contains CBOR data decoded from UR format using `GA_bcur_decode`.

```
{
  "ur_type": "crypto-seed",
  "data": "A20150C7098580125E2AB0981253468B2DBC5202D8641947DA"
}
```

ur_type The type of the decoded data as specified when it was encoded.

data The resulting CBOR-encoded data in hexadecimal format.

2.37 Settings JSON

Contains the users settings.

```
{
  "altimeout": 5,
  "csvtime": 51840,
  "nlocktime": 12960,
  "notifications": {
    "email_incoming": false,
    "email_login": false,
    "email_outgoing": false
  },
  "pricing": {
    "currency": "USD",
    "exchange": "BITSTAMP"
  },
  "required_num_blocks": 12,
  "sound": true,
  "unit": "BTC"
}
```

```
{
  "altimeout": 5,
  "pricing": {
    "currency": "USD",
    "exchange": "BITFINEX"
  },
  "required_num_blocks": 12,
  "sound": true,
  "unit": "BTC"
}
```

```
{
  "altimeout": 5,
  "csvtime": 65535,
  "nlocktime": 12960,
  "notifications": {
    "email_incoming": false,
    "email_login": false,
    "email_outgoing": false
  },
  "pricing": {
    "currency": "USD",
    "exchange": "BITSTAMP"
  },
  "required_num_blocks": 12,
  "sound": true,
  "unit": "BTC"
}
```

```
{
  "altimeout": 5,
  "pricing": {
    "currency": "USD",
    "exchange": "BITFINEX"
  },
  "required_num_blocks": 12,
  "sound": true,
  "unit": "BTC"
}
```

2.38 Receive address request JSON

Contains the query parameters for requesting an address using *GA_get_receive_address*.

```
{
  "subaccount": 0,
  "is_internal": false,
  "ignore_gap_limit": false,
}
```

subaccount Mandatory. The value of “pointer” from *Subaccounts list JSON* or *Subaccount JSON* for the subaccount to fetch addresses for.

is_internal Optional, singlesig only. Whether or not the user key belongs to the internal chain.

ignore_gap_limit Optional, singlesig only. Whether to allow sequentially generated addresses to go beyond the GAP_LIMIT. This is potentially dangerous as funds received on such addresses are not synced until an address within the GAP_LIMIT receives funds.

2.39 Receive address details JSON

```
{
  "address": "2N2zyKJEcAtqgHRdYFqCQDKZozDi5GY9wi5",
  "address_type": "csv",
  "branch": 1,
  "pointer": 1,
  "script":
  ↪ "2102ea2fe5fcee6e1ada9ee3f4228ac1b83554cff618c8bc6ef64d9988e2f4bb13a1ad21023cebb0d3a3434d1df033c3c",
  ↪ ",
  "script_type": 15,
  "scriptpubkey": "a9146aff2fc8f5ec359f809b3355d6bdc5d30f8e105487",
  "service_xpub":
  ↪ "tpubEATpVqYYmDa5DQdXdJhaMyrvycVbpuCRdUTVifuG3ML5kbwujP3CdxGT1rVoDnFnxsxEv1aiVaEtuJzDBulbNjK9mADe",
  ↪ ",
  "subaccount": 0,
  "subtype": 51840,
  "user_path": [
    1,
    1
  ]
}
```

```
{
  "address": "2MsHqxhz5fHt2QyvE8TgT1MvzDY9kdMH5Rz",
  "address_type": "p2sh-p2wpkh",
  "is_internal": false,
  "pointer": 1,
  "scriptpubkey": "a914007edc4b1153a320e490b5fe29a33e95af2d3c7387",
  "subaccount": 0,
  "user_path": [
    2147483697,
    2147483649,
    2147483648,
    0,
    1
  ]
}
```

(continues on next page)

(continued from previous page)

```
]
}
```

```
{
  "address":
  ↪ "AzpsXif2CBq9WEtyZW5JR85LUEBXjvZXi2wx5EaKqXQ1cQh8Jo35fGva7MWcfQmkNZ3D2hZbXD3U16eF",
  "address_type": "csv",
  "blinding_key": "033c13b87298d77941d6e4ec5c754a03a41a7c48ad4196e1e97a734d8646daa231
  ↪ ",
  "branch": 1,
  "is_confidential": true,
  "pointer": 1,
  "script":
  ↪ "748c6321023cebb0d3a3434d1df033c3c41e3d02f2e0b44a64c3d58519ca656b4f5363f591ad6703ffff00b275682102ea
  ↪ ",
  "script_type": 15,
  "scriptpubkey": "a91469a22b9f6314c0fe6db92286592ec287ac52c03587",
  "service_xpub":
  ↪ "tpubEATpVqYYmDa5DQdXdJhaMyrvycVbpuCRdUTVifuG3ML5kbwujP3CdxGT1rVoDnFnxsxEv1aiVaEtuJzDBulbNJk9mADe
  ↪ ",
  "subaccount": 0,
  "subtype": 65535,
  "unconfidential_address": "XLyn55q1TtektRH3YrDTT8vMTMkjrcT38u",
  "user_path": [
    1,
    1
  ]
}
```

```
{
  "address":
  ↪ "AzpjM6F3vhwPK3U5p8u79osjbhsduwdGCrbt1Cahy8nD3d7SbQJte99KiMkxWZLBrmUnWPrU3eLtUbVU",
  "address_type": "p2sh-p2wpkh",
  "blinding_key": "020de79e60d9d17b22c9f803455eb2fc48becc50fe76d32508bde357bf10c28a8a
  ↪ ",
  "is_confidential": true,
  "is_internal": false,
  "pointer": 1,
  "scriptpubkey": "a914007edc4b1153a320e490b5fe29a33e95af2d3c7387",
  "subaccount": 0,
  "unconfidential_address": "XBPPrpZvDkTp1TY3kAhNtLBzo4Vycb9BsQt",
  "user_path": [
    2147483697,
    2147483649,
    2147483648,
    0,
    1
  ]
}
```

address The wallet address in base58, bech32 or blech32 encoding.

address_type One of "csv", "p2sh", "p2wsh" (multisig), or "p2pkh", "p2sh-p2wpkh", "p2wpkh" (singlesig), indicating the type of address.

branch Always 1, used in the address derivation path for subaccounts.

pointer The address number/final number in the address derivation path.

script The locking script of the address.

scriptpubkey The scriptpubkey of the address.

script_type Integer representing the type of script.

subaccount The subaccount this address belongs to. Matches "pointer" from *Subaccounts list JSON* or *Subaccount JSON*.

subtype For "address_type" "csv", the number of CSV blocks referenced in "script", otherwise, 0.

user_path The BIP32 path for the user key.

For Liquid addresses, the following additional fields are returned:

```
{
  "blinding_key": "02a519491b130082a1abbe17395213b46dae43c3e1c05b7a3dbd2157bd83e88a6e",
  "is_blinded": true,
  "unconfidential_address": "XV4PaYgbaJdPnYaJDzE41TpbBF6yBieeyd"
}
```

blinding_key The blinding key used to blind this address.

is_blinded Always `true`.

unconfidential_address The non-confidential address corresponding to address. This is provided for informational purposes only and should not be used to receive.

2.40 Previous addresses request JSON

Contains the query parameters for requesting previously generated addresses using *GA_get_previous_addresses*.

```
{
  "subaccount": 0,
  "last_pointer": 0,
}
```

subaccount Mandatory. The value of "pointer" from *Subaccounts list JSON* or *Subaccount JSON* for the subaccount to fetch addresses for.

last_pointer The address pointer from which results should be returned. If this key is not present, the newest generated addresses are returned. If present, the "last_pointer" value from the resulting *Previous addresses JSON* should then be given, until sufficient pages have been fetched or the "last_pointer" key is not present indicating all addresses have been fetched.

is_internal Singlesig only. Whether or not the user key belongs to the internal chain.

2.41 Previous addresses JSON

Contains a page of previously generated addresses, from newest to oldest.

```

{
  "list": [
    {
      "address": "2N5oBDdNp9i3dyYXCEfimg4YEld9hcWYc2T",
      "address_type": "csv",
      "branch": 1,
      "pointer": 2,
      "script":
↪ "210251b4473cc52200be514994986927632cfe04b2fbe6fb1a69a0ea835dc1fe2cead210325d8aed10e04b73aab80b9e"
↪ ",
      "script_type": 15,
      "scriptpubkey": "a91489ac8209bdfdd8229d3123215e5eac3f859a6e4f87",
      "service_xpub":
↪ "tpubEATUpVqYYmDa5DQdXdJhaMyrvycVbpuCRdUTVifuG3ML5kbwujP3CdxGT1rVoDnFnxsxEvlaiVaEtuJzDBulbNJK9mADee"
↪ ",
      "subaccount": 0,
      "subtype": 51840,
      "tx_count": 0,
      "user_path": [
        1,
        2
      ]
    },
    {
      "address": "2N2zyKJEcAtqgHRdYFqCQDKZozDi5GY9wi5",
      "address_type": "csv",
      "branch": 1,
      "pointer": 1,
      "script":
↪ "2102ea2fe5fcee6e1ada9ee3f4228ac1b83554cff618c8bc6ef64d9988e2f4bb13a1ad21023cebb0d3a3434d1df033c3c"
↪ ",
      "script_type": 15,
      "scriptpubkey": "a9146aff2fc8f5ec359f809b3355d6bdc5d30f8e105487",
      "service_xpub":
↪ "tpubEATUpVqYYmDa5DQdXdJhaMyrvycVbpuCRdUTVifuG3ML5kbwujP3CdxGT1rVoDnFnxsxEvlaiVaEtuJzDBulbNJK9mADee"
↪ ",
      "subaccount": 0,
      "subtype": 51840,
      "tx_count": 0,
      "user_path": [
        1,
        1
      ]
    }
  ]
}

```

```

{
  "list": [
    {
      "address": "2MsHqxhz5fHt2QyvE8TgT1MvzDY9kdMH5Rz",
      "address_type": "p2sh-p2wpkh",
      "is_internal": false,
      "pointer": 1,
      "script": "a914007edc4b1153a320e490b5fe29a33e95af2d3c7387",
      "scriptpubkey": "a914007edc4b1153a320e490b5fe29a33e95af2d3c7387",
      "subaccount": 0,

```

(continues on next page)

(continued from previous page)

```

        "tx_count": 0,
        "user_path": [
            2147483697,
            2147483649,
            2147483648,
            0,
            1
        ]
    },
    {
        "address": "2N4uHLYgfrtRTPQYJe8DHT69WgsUTYe6WSo",
        "address_type": "p2sh-p2wpkh",
        "is_internal": false,
        "pointer": 0,
        "script": "a9147fdbfc6f9f1ecd1d66f43a586af08330a73e3cf487",
        "scriptpubkey": "a9147fdbfc6f9f1ecd1d66f43a586af08330a73e3cf487",
        "subaccount": 0,
        "tx_count": 0,
        "user_path": [
            2147483697,
            2147483649,
            2147483648,
            0,
            0
        ]
    }
]
}

```

```

{
    "list": [
        {
            "address":
↪ "AzpxKzziQPVEFaqqoTEofLfbDoCYopCs5LisZhbr56wTmkiTxaq4Ue4K9E6wBx6L9BvNUYaW8vobchSZ",
            "address_type": "csv",
            "blinding_key":
↪ "03ed3ed04c8e522af02a760ca8feb42ac8c298ee25d16718efce95dd530c59161c",
            "branch": 1,
            "is_confidential": true,
            "pointer": 2,
            "script":
↪ "748c63210325d8aed10e04b73aab80b9e738e3f3933d49de0238a4ff0a3395cd7f3e5ba65fad6703ffff00b27568210251",
↪ ",
            "script_type": 15,
            "scriptpubkey": "a914e5c73e6d8119f62b31f8c1b7069e36088c2779f387",
            "service_xpub":
↪ "tpubEATUpVqYYmDa5DQdXdJhaMyrvycVbpuCRdUTVifuG3ML5kbwujP3CdxGT1rVoDnFnxsxEvlaiVaEtuJzDBulbNJk9mADe",
↪ ",
            "subaccount": 0,
            "subtype": 65535,
            "tx_count": 0,
            "unconfidential_address": "XYJCGnCEYhRzPwPfFyk433jUHKzKhvwNiD",
            "user_path": [
                1,
                2
            ]
        }
    ]
}

```

(continues on next page)

(continued from previous page)

```

    },
    {
      "address":
↪ "AzpsXif2CBq9WEtyZW5JR85LUEBXjvZXi2wx5EaKqXQ1cQh8Jo35fGva7MWcfQmkNZ3D2hZbXD3U16eF",
      "address_type": "csv",
      "blinding_key":
↪ "033c13b87298d77941d6e4ec5c754a03a41a7c48ad4196e1e97a734d8646daa231",
      "branch": 1,
      "is_confidential": true,
      "pointer": 1,
      "script":
↪ "748c6321023cebb0d3a3434d1df033c3c41e3d02f2e0b44a64c3d58519ca656b4f5363f591ad6703ffff00b275682102e",
↪ "",
      "script_type": 15,
      "scriptpubkey": "a91469a22b9f6314c0fe6db92286592ec287ac52c03587",
      "service_xpub":
↪ "tpubEAUTpVqYYmDa5DQdXdJhaMyrvycVbpuCRdUTVifuG3ML5kbwujP3CdxGT1rVoDnFnxsxEvlaiVaEtuJzDBulbNJK9mADe",
↪ "",
      "subaccount": 0,
      "subtype": 65535,
      "tx_count": 0,
      "unconfidential_address": "XLyn55q1TtektRH3YrDTT8vMTMkjrcT38u",
      "user_path": [
        1,
        1
      ]
    }
  ]
}

```

```

{
  "list": [
    {
      "address":
↪ "AzpjM6F3vhwPK3U5p8u79osjbhsduwdGCrbt1Cahy8nD3d7SbQJte99KiMkxWZLBrmUnWPpU3eLtUbVU",
      "address_type": "p2sh-p2wpkh",
      "blinding_key":
↪ "020de79e60d9d17b22c9f803455eb2fc48becc50fe76d32508bde357bf10c28a8a",
      "is_confidential": true,
      "is_internal": false,
      "pointer": 1,
      "script": "a914007edc4b1153a320e490b5fe29a33e95af2d3c7387",
      "scriptpubkey": "a914007edc4b1153a320e490b5fe29a33e95af2d3c7387",
      "subaccount": 0,
      "tx_count": 0,
      "unconfidential_address": "XBPrpZvDkTp1TY3kAhNtLBzo4Vycb9BsQt",
      "user_path": [
        2147483697,
        2147483649,
        2147483648,
        0,
        1
      ]
    },
    {
      "address":
↪ "AzprjS3xmCowJRjRzQ+qufBjLtsHztgVAAEF3Z2ChTLrDb8jWEK3RX2yWLZnFZyvdHFYJVgtAMhOYfyCE",

```

(continues on next page)

(continued from previous page)

```

    "address_type": "p2sh-p2wpkh",
    "blinding_key":
↪ "031e9bdbe8710e94a18193f91a5bb9f22e13f84581a7b6d51db340b421907ee546",
    "is_confidential": true,
    "is_internal": false,
    "pointer": 0,
    "script": "a9147fdbfc6f9f1ecd1d66f43a586af08330a73e3cf487",
    "scriptpubkey": "a9147fdbfc6f9f1ecd1d66f43a586af08330a73e3cf487",
    "subaccount": 0,
    "tx_count": 0,
    "unconfidential_address": "XP1JCQcox4MSRxfgpMuimvDKXqJKWSxxWv",
    "user_path": [
        2147483697,
        2147483649,
        2147483648,
        0,
        0
    ]
}
]
}

```

last_pointer If present indicates that there are more addresses to be fetched, and the caller to get the next page should call again *GA_get_previous_addresses* passing this value in *Previous addresses request JSON*. If not present there are no more addresses to fetch.

list Contains the current page of addresses in *Receive address details JSON* format.

2.42 Unspent outputs request JSON

Describes which unspent outputs to return from *GA_get_unspent_outputs*, or which unspent outputs to include in the balance returned by *GA_get_balance*.

```

{
  "subaccount": 3,
  "num_confs": 0,
  "all_coins": false,
  "expired_at": 99999,
  "confidential": false,
  "dust_limit": 546,
  "sort_by": "newest"
}

```

subaccount The subaccount to fetch unspent outputs for.

num_confs Pass 0 for unconfirmed UTXOs or 1 for confirmed.

all_coins Pass `true` to include UTXOs with status `frozen`. Defaults to `false`.

expired_at If given, only UTXOs where two-factor authentication expires by the given block are returned.

confidential Pass `true` to include only confidential UTXOs. Defaults to `false`.

dust_limit If given, only UTXOs with a value greater than the limit value are returned.

sort_by One of "oldest", "newest", "largest", "smallest". Returns the unspent outputs sorted by block height or value respectively. If not given, defaults to "oldest" for 2of2 subaccounts and "largest" for other subaccount types.

2.43 Unspent outputs for private key request JSON

Describes the private key to search for unspent outputs for with *GA_get_unspent_outputs_for_private_key*.

```
{
  "private_key": "6PRK95NQL1rJWZYegfeY1x2vPdsWFSiDDJTzIatqkpVFeYi3osJDtiQiw9",
  "password": "foobar"
}
```

private_key Mandatory. The private key in WIF or BIP 38 format.

password Optional. The password the key is encrypted with, if any.

2.44 Unspent outputs JSON

Contains unspent outputs for the wallet as requested by *GA_get_unspent_outputs*.

```
{
  "unspent_outputs": {
    "btc": [
      {
        "address_type": "csv",
        "block_height": 0,
        "is_internal": false,
        "pointer": 1,
        "prevout_script":
        ↪ "2102ea2fe5fcee6e1ada9ee3f4228ac1b83554cff618c8bc6ef64d9988e2f4bb13a1ad21023cebb0d3a3434d1df033c3c",
        ↪ ",
        "pt_idx": 0,
        "satoshi": 12340,
        "script_type": 15,
        "subaccount": 0,
        "subtype": 51840,
        "txhash": "1a45e3a1140b17b45277e7e1189ff8385c7e0fd4cd88a5840b15ee600d5dbc30",
        "user_status": 0
      }
    ]
  }
}
```

```
{
  "unspent_outputs": {
    "btc": [
      {
        "address_type": "p2sh-p2wpkh",
        "block_height": 0,
        "is_internal": false,
        "pointer": 1,
        "prevout_script": "76a9144d2af219985cd7e523cf1be0421636fe4f38f4b788ac",

```

(continues on next page)

(continued from previous page)

```

        "pt_idx": 1,
        "public_key":
↪ "030e18cc1d6474e092264aa44e7b81ce1068f2b12ec748c96f217d8de5cb2555dc",
        "satoshi": 12340,
        "subaccount": 0,
        "txhash": "8a3974692b29237cb81f545a7891be89d73aed7d0038879da437623b0fd57584",
        "user_path": [
            2147483697,
            2147483649,
            2147483648,
            0,
            1
        ]
    }
}
}
}

```

```

{
  "unspent_outputs": {
    "5ac9f65c0efcc4775e0baec4ec03abdde22473cd3cf33c0419ca290e0751b225": [
      {
        "address_type": "csv",
        "amountblinder":
↪ "337fbb7a7c554a1a11afd58524a650811e6a2e655966e5e8870c6aa5bc220c3",
        "asset_id": "5ac9f65c0efcc4775e0baec4ec03abdde22473cd3cf33c0419ca290e0751b225"
↪ ",
        "asset_tag":
↪ "0aec3e557312bbe41d1ead85eb9e2a741965f5fb7426a398c882c8bb9b6eb0a1d9",
        "assetblinder":
↪ "0ea4abfc0c7c36929a6d051d9694ac58a5c57a571af89bf34337a02e29552b7a",
        "block_height": 0,
        "commitment":
↪ "09f9d304ce60ad52731ea60b64e069623c4df719062cc49f8d3a9e3ca06ef411b2",
        "is_blinded": true,
        "is_confidential": false,
        "is_internal": false,
        "nonce_commitment":
↪ "03b4ff371f4545ba5f1f9b58c6fea3ea99939c86f206ca594277a8d24d2d7333ed",
        "pointer": 1,
        "prevout_script":
↪ "748c6321023cebb0d3a3434d1df033c3c41e3d02f2e0b44a64c3d58519ca656b4f5363f591ad6703ffff00b275682102e"
↪ ",
        "pt_idx": 0,
        "satoshi": 12340,
        "script": "a91469a22b9f6314c0fe6db92286592ec287ac52c03587",
        "script_type": 15,
        "subaccount": 0,
        "subtype": 65535,
        "txhash": "46dd2e84591e3a27c85d3ead917a1469dd09458fb1d76bce6b58885bf5d6ebbc",
        "user_status": 0
      }
    ]
  }
}

```

```
{
  "unspent_outputs": {
    "5ac9f65c0efcc4775e0baec4ec03abdde22473cd3cf33c0419ca290e0751b225": [
      {
        "address_type": "p2sh-p2wpkh",
        "amountblinder":
        ↪ "8efb0d0b21e6f14960f4c2affc166e50555f03bfc5ece3f2992b6ab4fd975caf",
        "asset_id": "5ac9f65c0efcc4775e0baec4ec03abdde22473cd3cf33c0419ca290e0751b225
        ↪ ",
        "asset_tag":
        ↪ "0a552d971ca55f71e25de1f4961f0646445ec7233f060af79db1ed5d1b7944adc1",
        "assetblinder":
        ↪ "a986f588d8f05e337627bc780732efde594ad3cf32cb17e994dc7dcb25bc72fd",
        "block_height": 0,
        "commitment":
        ↪ "091a8f5a5542d635303eaa52cf6ed0cf32d084be85fe292266427dfb7e97277111",
        "is_blinded": true,
        "is_confidential": false,
        "is_internal": false,
        "nonce_commitment":
        ↪ "026500ef0c14658df948d9f99d76c37724377f9f66860c2d3b67bc47be4e763a3f",
        "pointer": 1,
        "prevout_script": "76a9144d2af219985cd7e523cf1be0421636fe4f38f4b788ac",
        "pt_idx": 1,
        "public_key":
        ↪ "030e18cc1d6474e092264aa44e7b81ce1068f2b12ec748c96f217d8de5cb2555dc",
        "satoshi": 12340,
        "subaccount": 0,
        "txhash": "eea0ccc14f823dfcf6afcaf9f57db43aac294d675dd19bb3b563fc11ee169864",
        "user_path": [
          2147483697,
          2147483649,
          2147483648,
          0,
          1
        ]
      }
    ]
  }
}
```

txhash The txid of the transaction.

pt_idx The index of the output, the vout.

satoshi The amount of the output.

block_height The height of the block where the transaction is included. Is 0 if the transaction is unconfirmed.

address_type One of "csv", "p2sh", "p2wsh" (multisig), or "p2pkh", "p2sh-p2wpkh", "p2wpkh" (singlesig), indicating the type of address.

is_internal Whether or not the user key belongs to the internal chain. Always false for multisig.

pointer The user key number/final number in the derivation path.

subaccount The subaccount this output belongs to. Matches "pointer" from *Subaccounts list JSON* or *Subaccount JSON*.

prevout_script The script being signed, the script code.

user_path The BIP32 path for the user key.

public_key Singlesig only. The user public key.

expiry_height Multisig only. The block height when two-factor authentication expires.

script_type Multisig only. Integer representing the type of script.

user_status Multisig only. 0 for "default" and 1 for "frozen".

subtype Multisig only. For "address_type" "csv", the number of CSV blocks referenced in "script", otherwise, 0.

For Liquid instead of having the "btc" field, there are (possibly) multiple fields, one for each asset owned, and the keys are the hex-encoded policy ids.

For Liquid the inner maps have additional fields:

amountblinder The hex-encoded amount blinder (value blinding factor, vbf).

asset_id The hex-encoded asset id in display format.

asset_tag The hex-encoded asset commitment.

assetblinder The hex-encoded asset blinder (asset blinding factor, abf).

commitment The hex-encoded value commitment.

is_blinded A boolean indicating whether or not the output is blinded.

nonce_commitment The hex-encoded nonce commitment.

2.45 Unspent outputs set status JSON

Valid status values are "default" for normal behavior or "frozen". Frozen outputs are hidden from the caller's balance and unspent output requests, are not returned in nlocktime emails, and cannot be spent. An account containing frozen outputs can be deleted, whereas an account with unfrozen outputs can not.

Freezing an output requires two factor authentication. Outputs should only be frozen in response to e.g. a dust attack on the wallet. Once a wallet is deleted, any frozen outputs it contained will be unspendable forever.

Note: Only outputs of value less than two times the dust limit can be frozen.

```
{
  "list" : [
    {
      "txhash": "09933a297fde31e6477d5aab75f164e0d3864e4f23c3afd795d9121a296513c0",
      "pt_idx": 1,
      "user_status": "frozen"
    }
  ]
}
```

2.46 Transactions details JSON

```
{ "subaccount":0, "first":0, "count":30 }
```

2.47 Network JSON

Contains the data describing a network the caller can connect to.

```
{
  "address_explorer_url": "https://blockstream.info/address/",
  "bech32_prefix": "bc",
  "bip21_prefix": "bitcoin",
  "blob_server_onion_url": "",
  "blob_server_url": "",
  "csv_buckets": [
    25920,
    51840,
    65535
  ],
  "development": false,
  "electrum_onion_url": "explorerzydxu5ecjrkwcayqybizmpjjznk5izmitf2modhcsuqlid.
↪ onion:110",
  "electrum_tls": true,
  "electrum_url": "blockstream.info:700",
  "liquid": false,
  "mainnet": true,
  "max_reorg_blocks": 144,
  "name": "Bitcoin",
  "network": "mainnet",
  "p2pkh_version": 0,
  "p2sh_version": 5,
  "pin_server_onion_url": "http://
↪ mrrxtq6tjpbnbm7vh5jt6mpjctn7ggfyf5wegvbef3x7jrnqawlmid.onion",
  "pin_server_public_key":
↪ "0332b7b1348bde8ca4b46b9dcc30320e140ca26428160a27bdbfc30b34ec87c547",
  "pin_server_url": "https://jadepin.blockstream.com",
  "price_onion_url": "http://greenv32e5p4rax6dmf4z17kq2fbmizd7miyava2actplmipyx2qd.
↪ onion/prices",
  "price_url": "https://green-bitcoin-mainnet.blockstream.com/prices",
  "server_type": "green",
  "service_chain_code":
↪ "e9a563d68686999af372a33157209c6860fe79197a4dafd9ec1dbaa49523351d",
  "service_pubkey":
↪ "0322c5f5c9c4b9d1c3e22ca995e200d724c2d7d8b6953f7b38fddf9296053c961f",
  "spv_enabled": false,
  "spv_multi": false,
  "spv_servers": [],
  "tx_explorer_url": "https://blockstream.info/tx/",
  "wamp_cert_pins": [
    "46494e30379059df18be52124305e606fc59070e5b21076ce113954b60517cda",
    "b42688d73bac5099d9cf4fdb7b05f5e54e98c5aa8ab56ee06c297a9a84d2d5f1",
    "67add1166b020ae61b8f5fc96813c04c2aa589960796865572a3c7e737613dfd",
    "64e286b76063602a372efd60cde8db2656a49ee15e84254b3d6eb5fe38f4288b"
  ],
  "wamp_cert_roots": [
    "\n-----BEGIN CERTIFICATE-----\nMIICxjCCAk2gAwIBAgIRAL093/
↪ inhFu86QOgQTWzSkUwCgYIKoZIzj0EAwMwTzEL\nMAkGA1UEBhMCVVMxKTAnBgNVBAoTIEludGVybmV0IFNlY3VyaXR5IFJlc2V
↪ xwLcA/TmC9N/A8AmzfwAVXMpcuBe8qYwJ\n+240JxP2T35p0wKZXuskr5LBJJvmsSGPwSSB/
↪ GjMH2m6WPUZlvd0xhajggEIMIIB\nBDAOBgNVHQ8BAf8EBAMCAYYwHQYDVR0LBBYwFAYIKwYBBQUHAwIGCCsGAQUFBwMB\nnMBI
↪ wQIMAYBAf8CAQAwHQYDVR0OBBYEFFrz7Sv8NsI3eblSMOpUb89V\nnyy6sMB8GA1UdIwQYMBaFAFHxC1q7eS0g7+pL4nozPbYupc
↪ AwCd89eYhN1XLr3pD5xcSAQh8suzYHK019YST8pE9kLJ03uGqQIwWrGxtO3q\nnYJkgsTgDyJ2gJrjubi1K9sZmHzOa25JK1fUp
↪ Lgu1\n-----END CERTIFICATE-----",
```

(continues on next page)

(continued from previous page)

```

"\n-----BEGIN CERTIFICATE-----\nMIICxjCCAkYgAwIBAgIQTtI99q9+x/
→mwXHJv+VEqdzAKBggqhkJOPQDZBPMQsw\nCQYDVQQGEwJVUzEpMCcGA1UEChMgSW50ZXJuZXQgU2VjdXJpdHkgUmVzZWZyY2
→LbBVDxCsnsI7vuf4xp4Cu+ETNggGE\nyBqJ3j8iUwe5Yt/qfSrRf1/
→D5R58duaJ+IvLRXeASRqEL+VkdXrW3qOCAQgwggEE\nnMA4GA1UdDwEB/
→wQEAwIBhjAdBgNVHSUEFjAUBgggrBgEFBQcDAGYIKWYBBQUHAAwEw\nnEgYDVDR0TAQH/BAGwBgEB/
→wIBADAdBgNVHQ4EFgQUbZkq9U0C6+MRwWC6km+NPS7x\nn6kQwHwYDVR0jBBGwFoAUfEKWrt5LSdv6kviejm9t161yN5UwMgYIK
→No+KQ\nnUAM1u34E36neL/Zba7ombkIOchSgx1iVxzqtFWGddgoG+tpRPWhuhhn\nn-----END
→CERTIFICATE-----",
"\n-----BEGIN CERTIFICATE-----
→\nMIIFWjCCAOKgAwIBAgIQbkepxUtHDA3sM9CJuRz04TANBgkqhkiG9w0BAQwFADBH\nnMQswCQYDVQQGEwJVUzEiMCAGA1UEChM
→vo27xqLpeXo4xL+Sv2sfnOhB2x+cWX3u+58qPpvBKJXqeqUqv4IyflPLGcY9vX\nnmX7wC17raKb0xlpHdU0QM+NoSROjyBhsS+
→SCYe7\nnzUjwTcLCeoiKu7rPWRnWr4+wB7CeMfGCwcDfLqZtbBkOtdh+JhpFAz2weaSUKKOP\nnfyblqAj+lug8aJRT7oM6iCsV1
→kB+Xj9e1x3+naH+uzfsQ551Ve0vSbvlGHR6xYKu44LtcXFilWr06zqkUsp\nnzBmkMiVOKvFlRNACzqrOSbTqn3yDsEB750Orp2
→VjsPOS+C12LOO\nnRc92wO1AK/
→1TD7Cn1TsNsYqiA94xrcx36m97PtbfkSIS5r762DL8EGMUUXLeXdyW\nnk70paDPvOmsB4om3xPXV2V4J95eSRQAogB/
→mqgghtqmxlbCluQ0WEdrHbEg8QOB+\nndVrNVjzRlW5y0vtOUucD/SVRNuJLDWcfr0wbrM7Rv1/
→oFB2ACYPTTrIrnqYNxgF\nnlQIDAQABO0IwQDAOBGNVHQ8BAF8EBAMCAQYwDwYDVR0TAQH/BAUwAwEB/
→zAdBgNV\nnHQ4EFgQU5K8rJnEaK0gnhS9Sizv8IkTcT4wDQYJKoZIhvcNAQEMBAQDggIBADiW\nnCu49tJYeX++dnAsznyvgv3
→UEBZvXw6Z\nnXPYfcX3v73svfu02lpdwCxu11xWaj0140k4DLh9+42FpLFZXvRq4d2h9mREruZR\nngyFmxhE+885H7pwoHyXa/
→6xmld01D1zvICxi/ZG6qcZ8WpyTgYmpl0p8WnK0OdC3\nnd8t5/
→Wk6kjftbjhlRn7pYL15iJdf0BL07q9bgsiG1eGZbYwE8na6SfZu6W0eX6Dv\nnJ4J2QPim01hcDyx2kLGe4g0x8HYRZvBPsVhH
→Zg\nnDdIx02OYI5NaAIFiTo/
→Nis3Jz5nu2Z6qNuFoS3FJFYDo0j0dzpqPJeaAcWErtXvM\nn+SUWgeExX6GjfhaknBZqlxi9dnK1C54dNuYvoS++cJEPqOba+MS
→ZRBiIs6Kl0obsXs7X9\nnSQ98POyDGCBDTtWTurQ0sR8WNh8M5mq5Fkzc4P4dyKliPUDqysU0ArSuiYgzNdws\nnE3PYJ/
→HQcu51OyLemGhmW/HGY0dVHLqlCFF1pkgl\nn-----END CERTIFICATE-----",
"\n-----BEGIN CERTIFICATE-----
→\nMIIFazCCA0GwAwIBAgIRAIQz7DSQONZRGpGu2OCiwAwDQYJKoZIhvcNAQELBQAw\nnTzELMAKGA1UEBhMCVVMxKTAnBgNVBA
→vVqbvYATyjb3miGbESTtrFj/RQSa78f0uoxmyF+\nnoTM8ukj13Xnfs7j/
→EvEhmkvBioZxaUpmZmyPfjxwv60pIgbz5MDmgK7iS4+3mX6U\nnA5/
→TR5d8mUgJ+g4rk8Kb4Mu0U1XjIB0ttov0DiNewNwIRt18jA8+o+u3dpjq+sW\nnT8KOEut+zwvo/
→7V3LvSye0rgTBIldHCNAymg4VMk7BPZ7hm/
→ELNKjD+Jo2FR3qyH\nnB5T0Y3HSLuJvW5iB4Y1cNHlsdu87kGJ55tukmi8mxdAQ4Q7e2RCOFvu396j3x+UC\nnB5iPNgiV5+I3lg
→1JBdiB3QW0KtZB6awBdpUKD9jflb0SHzUv\nnKBds0pjBqAlkd25HN7rOrFleaJ1/
→ctaJxQZBKT5ZPt0m9STJEadao0xAH0ahmbWn\nnO1FuhjuefXKNEgV4We0+UXgVCwOPjdAvBbI+e0ocS3MFEVzG6uBQE3xDk3Sz
→rOPNk3sgrDQoo//fb4hVC1CLQJ13hef4Y53CI\nnrU7m2Ys6xt0nUW7/
→vGT1M0NPAGMBAAGjQjBAMA4GA1UdDwEB/wQEAwIBBjAPBgNV\nnHRMBAF8EBTADAQH/
→MB0GA1UdDgQWBRR5tFnme7b15AFzgAiIyBpY9umbbjANBgkq\nnhkiG9w0BAQsFAAOCAgEAVR9YqbyyqFDQDLHYGmkGjYkIrGF1
→V91ZL\nnubhzEFnTIZd+50xx+7LSYK05qAvqFyFWhfFQDlnrzuBZ6brJFe+GnY+EgPbk6ZGQ\nn3BebYhtF8GaV0nxvwuo77x/
→Py9auJ/GpsMiu/X1+mvoiBOv/2X/qkSsisRcOj/
→KK\nnNFtY2PwByVS5UcBmiogziUwthDyC3+6WVwW6LLv3xLfHTjuCvJHIInNzktHCgKQ5\nnORAZI4JMPJ+GslWYHb4phowim57ia
→+sKAIuvtd7u+Nxe5AW0wdeRlN8NwdC\nnjNPElpzVmbUq4JUagEiuTDkHszxHpFKVK7q4+63SM1N95R1NbdWhscdCb+ZAJzVc\nn
→1lvh+wjChP4kqK0J2qxq\nn4RgqsahDYvTh9w7jXbyLeiNdd8XM2w9U/t7y0Fff/
→9yi0GE44Za4rF2LN9d11TPA\nnmRGUnUHBcnWEvgJBQ19nJEiU0Zsnvgc/
→ubhPgXRR4Xq37Z0j4r7g1SgEEzwxAS7d\nnemyPxcgYxn/er44/KJ4EBs+1VDR3veyJm+kXQ99b21/
→+jh5Xos1AnX5iItreGCC=\nn-----END CERTIFICATE-----"
],
"wamp_onion_url": "ws://greenv32e5p4rax6dmfgeb4zz17kq2fbmizd7miyava2actplmipyx2qd.
→onion:80/v2/ws",
"wamp_url": "wss://green-bitcoin-mainnet.blockstream.com/v2/ws"
}

```

2.48 Network list JSON

Contains details of all available networks the API can connect to.

For each network listed, a *Network JSON* element is present containing the networks information.

```
{
  "all_networks": [
    "mainnet",
    "liquid",
    "testnet",
    "testnet-liquid",
    "electrum-liquid",
    "electrum-localtest",
    "electrum-localtest-liquid",
    "electrum-mainnet",
    "electrum-testnet",
    "electrum-testnet-liquid",
    "localtest",
    "localtest-liquid"
  ],
  "electrum-liquid": {
    "address_explorer_url": "https://blockstream.info/liquid/address/",
    "asset_registry_onion_url": "http://
↪lhquhzzpzg5tyymcqep24fynpzzqgg3m3rlh7ascnw5cpqsro35bfxyd.onion",
    "asset_registry_url": "https://assets.blockstream.info",
    "bech32_prefix": "ex",
    "bip21_prefix": "liquidnetwork",
    "blech32_prefix": "lq",
    "blinded_prefix": 12,
    "blob_server_onion_url": "",
    "blob_server_url": "",
    "csv_buckets": [],
    "development": false,
    "electrum_onion_url": "explorerzydxu5ecjrkwcayqybizmpjjznk5izmitf2modhcsuqlid.
↪onion:195",
    "electrum_tls": true,
    "electrum_url": "blockstream.info:995",
    "liquid": true,
    "mainnet": true,
    "max_reorg_blocks": 2,
    "name": "Liquid (Electrum)",
    "network": "electrum-liquid",
    "p2pkh_version": 57,
    "p2sh_version": 39,
    "pin_server_onion_url": "http://
↪mrrxtq6tjpbnbm7vh5jt6mpjctn7ggyfy5wegvbeff3x7jrznqawlmid.onion",
    "pin_server_public_key":
↪"0332b7b1348bde8ca4b46b9dcc30320e140ca26428160a27bdbfc30b34ec87c547",
    "pin_server_url": "https://jadepin.blockstream.com",
    "policy_asset": "6f0279e9ed041c3d710a9f57d0c02928416460c4b722ae3457a11eec381c526d
↪",
    "price_onion_url": "http://
↪greenv32e5p4rax6dmfqb4zzl7kq2fbmizd7miyava2actplmipyx2qd.onion/prices",
    "price_url": "https://green-bitcoin-mainnet.blockstream.com/prices",
    "server_type": "electrum",
    "service_chain_code": "",
    "service_pubkey": "",
    "spv_enabled": false,
    "spv_multi": false,
    "spv_servers": [],
    "tx_explorer_url": "https://blockstream.info/liquid/tx/",
  }
}
```

(continues on next page)

(continued from previous page)

```

    "wamp_cert_pins": [],
    "wamp_cert_roots": [
        "\n-----BEGIN CERTIFICATE-----\nMIICxjCCAk2gAwIBAgIRAL093/
        →inhFu86QOgQTWzSkUwCgYIKoZIzj0EAwMwTzEL\nMAkGA1UEBhMCVVMxKTAnBgNVBAoTIEludGVybmV0IFNlY3VyaXR5IFJlc2V
        →xwLCA/TmC9N/A8AmzfwdAVXMpcuBe8qQyWj\n+240JxP2T35p0wKZXusKR5LBJJvmsSGPwSSB/
        →GjMH2m6WPUZIVd0xhaJggEIMIIB\nBDAOBgNVHQ8BAf8EBAMCAYYwHQYDVR01BBYwFAYIKwYBBQUHAWIGCCsGAQUFBwMB\nnMBI
        →wQIMAYBAf8CAQAwwHQYDVR0OBBYEFFrz7Sv8NsI3eb1SMOpUb89V\nnyy6sMB8GA1UdIwQYMBaAFHxClq7eS0g7+pL4nozPbYupc
        →AwCd89eYhNlXlr3pD5xcSAQh8suzYHKOl9YST8pE9kLJ03uGqQIwWrGxtO3q\nyYJkgsTgDyJ2gJrjubilK9sZmHzOa25JK1fUpl
        →Lgul/\n-----END CERTIFICATE-----",
        "\n-----BEGIN CERTIFICATE-----\nMIICxjCCAkYgAwIBAgIQTtI99q9+x/
        →mwxHJv+VEqdzAKBggqhkjOPQQDAzBPMQsw\nnCQYDVQQGEwJVUzEpMCCGA1UEChMGSW50ZXJuZXQgU2VjdXJpdHkgUmVzZW
        →LbBVDxCSnsI7vuf4xp4Cu+ETNggGE\nnyBqJ3j8iUwe5Yt/qfSrRf1/
        →D5R58duaJ+IvLRXeASRqEL+VkDXrW3qOCAQgwgwEE\nnMA4GA1UdDwEB/
        →wQEAWIBhJAdBgNVHSUEFjAUBgggrBgEFBQcDAgYIKwYBBQUHAWAwEw\nnEgYDVR0TAQH/BAgwBgEB/
        →wIBADAdBgNVHQ4EFgQUbZkq9U0C6+MRwWC6km+NPS7x\nn6kQwHwYDVR0jBBgwFoAUfEKWrt5LSDv6kvieJm9ti6lyN5UwMgYIKw
        →No+KQ\nnUAMlu34E36neL/Zba7ombkIOchSgx1iVxzqtFWGddgoG+tpPRPWhuhhn\nn-----END
        →CERTIFICATE-----",
        "\n-----BEGIN CERTIFICATE-----
        →\nMIIFWjCCAOKgAwIBAgIQbkepxUtHDA3sM9CJuRz04TANBgkqhkiG9w0BAQwFADBH\nnMQswCQYDVQQGEwJVUzEiMCAgA1UEChM
        →vo27xqLpeXo4xL+Sv2sfnOhB2x+cWX3u+58qPpvBKJXqeqUqv4IyflPLGcY9vX\nnmX7wCl7raKb0xlpHdu0QM+NOsRojYbhs+
        →SCYe7\nnzUjwTcLceoiKu7rPWRnWr4+wB7CeMfGCwcdFlqZtbBkOtdh+JhpFAz2weaSUKKOP\nnfylblqAj+lug8aJRT7oM6iCsVlq
        →kB+Xj9elx3+naH+uzfsQ55lVe0vSbvlGHR6xYku44LtcXFilWr06zqkUsp\nnzBmkMiVOKvFlRNACzqrOSbTqn3yDsEB7500rp2y
        →VjsPOS+C12LOO\nnRc92w01AK/
        →1TD7Cn1TsNsYqiA94xrcx36m97PtbfkSIS5r762DL8EGMUUXLeXdyW\nnk70paDPvOmsB4om3pXV2V4J95eSRQAogB/
        →mqghtqmxlbCluQ0WEdrHbEg8QOB+\nndVrNVjzrLwW5y0vtOUucxD/SVRNuJLDWcfr0wbrM7Rv1/
        →oFB2ACYPTIrInqYnXgF\nnlQIDAQABO0IwQDAOBgNVHQ8BAf8EBAMCAQYwDwYDVR0TAQH/BAUwAwEB/
        →zAdBgNV\nnHQ4EFgQU5K8rJnEaK0gnhS9SZizv8IkTcT4wDQYJKoZIhvcNAQEMBQADggIBADiW\nnCu49tJYeX++dnAsznyvgvy3S
        →UEBzvXw6Z\nnXPYfcX3v73svfu02lpdwCxu11xWaj0140k4DLh9+42FpLFZXvRq4d2h9mREruZR\nngyFmxhE+885H7pwoHyXa/
        →6xnlD01D1zvICxi/ZG6qcz8WpyTgYmp10p8WnK00dC3\nnd8t5/
        →Wk6kjtftbjhlRn7pYL15iJdfOBL07q9bgsiG1eGZbYwE8na6SfZu6W0eX6Dv\nnJ4J2QPim01hcDyx2kLGe4g0x8HYRZvBPsVhH
        →Zg\nnDdIX02OYI5NaAIFItO/
        →Nis3Jz5nu2Z6qNuFoS3FJFDYo0j0dzpqPJeaAcWErtXvM\nn+SUWgeExX6GjfhaknBZqlxi9dnKlC54dNuYvoS++cJEPqOba+MS
        →ZRB1ls6Kl0obsXs7X9\nnsQ98POyDGCBDTtWTurQ0sR8WNh8M5mQ5Fkzc4P4dyKliPUDqysU0ArSuiYgzNdwS\nnE3PYJ/
        →HQcu51OyLemGhmW/HGY0dVHLqlCFF1pkgl\nn-----END CERTIFICATE-----",
        "\n-----BEGIN CERTIFICATE-----
        →\nMIIFazCCA1OgAwIBAgIRAIQz7DSQONZRGpGu2OCiAwAdQYJKoZIhvcNAQELBQAw\nnTzELMAkGA1UEBhMCVVMxKTAnBgNVBA
        →vVqbvYATyjb3miGbESTtrFj/RQSa78f0uoxmyF+\nn0TM8ukj13Xnfs7j/
        →EvEhmkvBioZxaUpmZmyPfjxwv60pIgbz5MDmgK7iS4+3mX6U\nnA5/
        →TR5d8mUgju+g4rk8Kb4Mu0U1XjIB0ttov0DiNewNwIRt18jA8+o+u3dpjq+sW\nnt8KOEut+zwvo/
        →7V3LvSye0rgTB1LDHCNAymg4VMk7BPZ7hm/
        →ELNKjD+Jo2FR3qyH\nnB5TOY3HsLuJvW5iB4Y1cNHlsdu87kGJ55tukmi8mxdAQ4Q7e2RCOFvu396j3x+UC\nnB5iPNgiV5+I3lq
        →1JBdiB3QW0KtZB6awBdpUKD9jflb0SHzUv\nnKBds0pjBqAlkd25HN7rOrFleaJ1/
        →ctaJxQZBKT5ZPt0m9STJEadao0xAH0ahmbWn\nnOlFuhjuefXKNEgV4We0+UXgVCwOPjdAvBbI+e0ocS3MFEVzG6uBQE3xDk3Szy
        →rOPNk3sgrDQoo//fb4hVC1CLQJ13hef4Y53CI\nnrU7m2Ys6xt0nUW7/
        →vGT1M0NPAGMBAAGjQjBAMA4GA1UdDwEB/wQEAWIBBjAPBgNV\nnHRMBAf8EBTADAQH/
        →MBOGA1UdDgQWBRR5tFnme7b15AFzgAiIyBpY9umbbjANBgkq\nnhkiG9w0BAQsFAAOCAgEAVR9YqbyyqfDQDLHYGmkGjYkIrGF12
        →V91ZL\nnubhzEFnTIZd+50xx+7LSYK05qAvqFyFWWhFQDlnrzuBZ6brJFe+GnY+EgPbk6ZGQ\nn3BebYhtF8GaV0nxvwuo77x/
        →Py9auJ/GpsMiu/X1+mvoiBOv/2X/qkSsisRcOj/
        →KK\nnNftY2PwByVS5uCbMiogziUwthDyC3+6WVwW6LLv3xLfHTjuCvJHIInNzktHCgKQ5\nnORAZi4JMPJ+GslWYHb4phowim57ia
        →+sKAIuvtd7u+Nxe5AW0wdeR1N8NwdC\nnjNPElpzVmbUq4JUagEiuTDkHszsHpFKVK7q4+63SM1N95R1NbdWhscdCb+ZAJzVc\nn
        →1lvh+wjChP4kqKOJ2qxq\nn4RgqsahDYVvTH9w7jXbyLeiNdd8XM2w9U/t7y0Ff/
        →9yi0GE44Za4rF2LN9d11TPA\nnmRGunUHBcnWEvgJBQ19nJEiU0Zsnvgc/
        →ubhPgXRR4Xq37Z0j4r7g1SgEEzwxA57d\nnemyPxgcYxn/eR44/KJ4EBs+lVDR3veyJm+kXQ99b21/
        →+jh5Xos1AnX5iItreGCC=\nn-----END CERTIFICATE-----"
    ],
    "wamp_onion_url": "",
    "wamp_url": ""

```

(continues on next page)

(continues on next page)

(continued from previous page)

```

"\n-----BEGIN CERTIFICATE-----
↪ \nMIIFazCCA1OgAwIBAgIRAIQz7DSQONZRGPGu2OCiwAwDQYJKoZIhvcNAQELBQAw\nTzELMAkGA1UEBhMCVVMxKTAnBgNVBA
↪ vVqbnvYATyjb3miGbESTtrFj/RQSa78f0uoxmyF+\n0TM8ukj13Xnfs7j/
↪ EvEhmkvBioZxaUpmZmyPfjxwv60pIgbz5MDmgK7iS4+3mX6U\nnA5/
↪ TR5d8mUgjU+g4rk8Kb4Mu0U1XjIB0ttov0DiNewNwIRt18jA8+o+u3dpjq+sW\nnT8KOEUt+zwvo/
↪ 7V3LvSye0rgTBiLDHCNAymg4VMk7BPZ7hm/
↪ ELNKjD+Jo2FR3qyH\nb5T0Y3HsLuJvW5iB4Y1cNHlsdu87kGJ55tukmi8mxdAQ4Q7e2RCOFvu396j3x+UC\nb5iPNgiV5+I31g
↪ lJBdiB3QW0KtZB6awBdpUKD9jflb0SHzUv\nKBds0pjBqAlkd25HN7rOrFleaJ1/
↪ ctaJxQZBKT5ZPt0m9STJEadao0xAH0ahmbWn\nOlFuhjuefXKNEgV4We0+UXgVCwOPjdAvBbI+e0ocS3MFEVzG6uBQE3xDk3Szy
↪ rOPNk3sgrDQoo//fb4hVC1CLQJ13hef4Y53CI\nnrU7m2Ys6xt0nUW7/
↪ vGT1M0NPAGMBAAGjQjBAMA4GA1UdDwEB/wQEAWIBBjAPBgNV\nnHRMBAf8EBTADAQH/
↪ MB0GA1UdDgQWBRR5tFnme7b15AFzgAiIyBpY9umbbjANBgkq\nnhkiG9w0BAQsFAAOCAGEA VR9YqbyyqFDQDLHYGmkGJykIrGF1
↪ V91ZL\nubhzEFnTIZd+50xx+7LSYK05qAvqFyFWhfQDlnrzuBZ6brJFe+GnY+EGPbk6ZGQ\nn3BebYhtF8GaV0nxvwuo77x/
↪ Py9auJ/GpsMiu/X1+mvoiBOv/2X/qkSsisRcOj/
↪ KK\nnFtY2PwByVS5UcBmiogziUwthDyC3+6WVwW6LLv3xLfHTjuCvJHIInNzktHCgKQ5\nnORAZI4JMPJ+GslWYHb4phowim57ia
↪ +sKAiuvtd7u+Nxe5AW0wdeRlN8NwdC\nnjNPElpzVmbUq4JUagEiuTDkHszsHpFKVK7q4+63SM1N95R1NbdWhscdCb+ZAJzVc\nn
↪ l1vh+wjChP4kqKOJ2qxq\nn4RgqsahDYVvTH9w7jXbyLeiNdd8XM2w9U/t7y0Ff/
↪ 9yi0GE44Za4rF2LN9d11TPA\nnmRGunUHBcnWEvgJBQ19nJEiU0Zsnvgc/
↪ ubhPgXRR4Xq37Z0j4r7g1SgEEzwxA57d\nnemyPxgcYxn/eR44/KJ4EBs+lVDR3veyJm+kXQ99b21/
↪ +jh5Xos1AnX5iItreGCc=\n-----END CERTIFICATE-----"
    ],
    "wamp_onion_url": "",
    "wamp_url": ""
  },
  "electrum-localtest-liquid": {
    "address_explorer_url": "",
    "asset_registry_onion_url": "http://
↪ lhquhzzpzg5tyymcqep24fynpzzqgg3m3rlh7ascnw5cpqsro35bfxyd.onion",
    "asset_registry_url": "https://assets.blockstream.info",
    "bech32_prefix": "ert",
    "bip21_prefix": "liquidnetwork",
    "blech32_prefix": "el",
    "blinded_prefix": 4,
    "blob_server_onion_url": "",
    "blob_server_url": "",
    "csv_buckets": [],
    "development": true,
    "electrum_onion_url": "",
    "electrum_tls": false,
    "electrum_url": "localhost:19002",
    "liquid": true,
    "mainnet": false,
    "max_reorg_blocks": 2,
    "name": "Localtest Liquid (Electrum)",
    "network": "electrum-localtest-liquid",
    "p2pkh_version": 235,
    "p2sh_version": 75,
    "pin_server_onion_url": "http://
↪ mrrxtq6tjpbnbm7vh5jt6mpjctn7ggyfy5wegvbef3x7jrznqawlmid.onion",
    "pin_server_public_key":
↪ "0332b7b1348bde8ca4b46b9dcc30320e140ca26428160a27bdbfc30b34ec87c547",
    "pin_server_url": "https://jadepin.blockstream.com",
    "policy_asset": "5ac9f65c0efcc4775e0baec4ec03abdde22473cd3cf33c0419ca290e0751b225
↪ ",
    "price_onion_url": "",
    "price_url": "http://localhost:8081",
    "server_type": "electrum",

```

(continues on next page)

(continued from previous page)

```

"service_chain_code": "",
"service_pubkey": "",
"spv_enabled": false,
"spv_multi": false,
"spv_servers": [],
"tx_explorer_url": "",
"wamp_cert_pins": [],
"wamp_cert_roots": [
    "\n-----BEGIN CERTIFICATE-----\nMIICXjCCAk2gAwIBAgIRAL093/
    →inhFu86QOgQTWzSkUwCgYIKoZIzj0EAwMwTzEL\nMAkGA1UEBhMCVVMxKTAnBgNVBAoTIEludGVybmV0IFNlY3VyaXR5IFJlc2
    →xwLCA/TmC9N/A8AmzfdAVXMpcuBe8qQyWj\n+240JxP2T35p0wKZXusKR5LBJJvmsSGPwSSB/
    →GjMH2m6WPUZiVd0xhajggEIMIIB\nBDAOBgNVHQ8BAf8EBAMCAYYwHQYDVR01BBYwFAYIKwYBBQUHAWIGCCsGAQUFBwMB\n    →wQIMAYBAf8CAQAwHQYDVR0OBBYEFFrz7Sv8NsI3eb1SM0pUb89V\nnyy6sMB8GA1UdIwQYMBaAFHxC1q7eS0g7+pL4nozPbYupc
    →AwCd89eYhN1XLr3pD5xcSAQh8suzYHK019YST8pE9kLJ03uGqQIwWrGxtO3q\nyYJkgsTgDyJ2gJrjubi1K9sZmHzOa25JK1fUp
    →Lgul/\n-----END CERTIFICATE-----",
    "\n-----BEGIN CERTIFICATE-----\nMIICXjCCAkYgAwIBAgIQTtI99q9+x/
    →mwXhJjv+VEqdzAKBggqhkjOPQQDAzBPMQsw\nnCQYDVQQGEwJVUzEpMCCGA1UEChMgSW50ZXJ1ZXQgU2VjdXJpdHkgUmVzZWZyY2
    →LbBVDxCSnsI7vuf4xp4Cu+ETNggGE\nnyBqJ3j8iUwe5Yt/qfSrRf1/
    →D5R58duaJ+IvLRXeASRqEL+VkDXrW3qOCAQgwggEE\nnMA4GA1UdDwEB/
    →wQEAWIBhjAdBgNVHSUEFjAUBgggrBgEFBQcDAgYIKwYBBQUHAWew\nnEgYDVR0TAQH/BAgwBgEB/
    →wIBADAdBgNVHQ4EFgQUbZkq9U0C6+MRwWC6km+NPS7x\nn6kQwHwYDVR01JBGwFoAUfEKWrt5LSDv6kviejm9ti6lyN5UwMgYIKv
    →No+KQ\nnUAM1u34E36neL/Zba7ombkIOchSgx1iVxztqFWGddgoG+tpPRPWhuhhn\n-----END
    →CERTIFICATE-----",
    "\n-----BEGIN CERTIFICATE-----
    →\nMIIFWjCCAOKgAwIBAgIQbkepxUtHDA3sM9CJuRz04TANBgkqhkiG9w0BAQwFADBH\nnMQswCQYDVQQGEwJVUzEiMCAGA1UEChM
    →vo27xqLpeXo4xL+Sv2sfnohB2x+cWX3u+58qPpvBKJXqeqUqv4IyflpLgCY9vX\nnmX7wCl7raKb0xlpH DU0QM+NoSROjyBhsS+
    →SCYe7\nnzUjwTcLceoiKu7rPWRnWr4+wB7CeMfGCwcdLfQZtbBkOtdh+JhpFaz2weaSUKK0P\nnfylblqAj+lug8aJRT7oM6iCsVlq
    →kB+Xj9elx3+naH+uzfsQ55lVe0vSbvlghR6xYKu44LtcXFilWr06zqkUsp\nnzBmkMiVOKvF1RNACzqrOSbtqn3yDsEB7500rp2y
    →VjsPOS+C12LOO\nnRc92w01AK/
    →1TD7Cn1TsNsYqiA94xrcx36m97PtbfkSIS5r762DL8EGMUUXLeXdYW\nnk70paDPvOmbS4om3xPXV2V4J95eSRQAogB/
    →mqghtqmxlbCluQ0WEdrHbEg8QOB+\nndVrNVjzRlw5y0vtOUucD/SVRNuJLDWcfr0wbrM7Rv1/
    →oFB2ACYPTIrInqYXngF\nnlQIDAQAB0IwQDAOBgNVHQ8BAf8EBAMCAQYwDwYDVR0TAQH/BAUwAwEB/
    →zAdBgNV\nnHQ4EFgQU5K8rJnEaK0gnhS9SZizv8IkTcT4wDQYJKoZIhvcNAQEMBQADggIBADiW\nnCu49tJYeX++dnAsznyvgyv3S
    →UEBZvXw6Z\nnXPYfcX3v73svfu021pdwCxu11xWaJ0140k4DLh9+42FpLFZxvRq4d2h9mREruZR\nngyFmxhE+885H7pwoHyXa/
    →6xmlld01D1zvICxi/ZG6qcz8WpyTgYmpl0p8WnK00dC3\nnd8t5/
    →Wk6kjftbjhlRn7pYL15iJdfOBL07q9bgsiG1eGZbYwE8na6SfZu6W0eX6Dv\nnJ4J2QPim01hcDyx2kLGe4g0x8HYRZvBPsVhH
    →Zg\nnDdIX02OYI5NaAIFItO/
    →Nis3Jz5nu2Z6qNuFoS3FJFDYoOj0dzpqPJeaAcWErtXvM\nn+SUWgeExX6GjfhaknBZqlxi9dnK1C54dNuYvoS++cJEPqOba+MS
    →ZRBils6K10obsXs7X9\nnSQ98POyDGCBDTtWTurQ0sR8WNh8M5mQ5Fkzc4P4dyKliPUDqysU0ArSuiYgzNdws\nnE3PYJ/
    →HQcu51OyLemGhmW/HGY0dVHLq1CFF1pkl\nn-----END CERTIFICATE-----",
    "\n-----BEGIN CERTIFICATE-----
    →\nMIIFazCCA1OgAwIBAgIRAIQz7DSQONZRGpGu2OCiAwDQYJKoZIhvcNAQELBQAw\nnTzELMAkGA1UEBhMCVVMxKTAnBgNVBAoTIEludGVybmV0IFNlY3VyaXR5IFJlc2
    →vVqbvYATyjb3miGbESTtrFj/RQSa78f0uoxmyF+\nn0TM8ukj13Xnfs7j/
    →EvEhmkvBioZxaUpmZmyPfjxwv60pIgbz5MDmgK7iS4+3mX6U\nnA5/
    →TR5d8mUgju+g4rk8Kb4Mu0U1XjIB0ttov0DiNewNwIRt18jA8+o+u3dpjq+sW\nnT8KOEut+zwvo/
    →7V3LvSye0rgTB1LDHCNAymg4VMK7BPZ7hm/
    →ELNKjD+Jo2FR3qyH\nnB5TOY3HsLuJvW5iB4Y1cNHlsdu87kGJ55tukmi8mxdAQ4Q7e2RCOFvu396j3x+UC\nnB5iPNgiV5+I31g
    →1JBdiB3QW0KtZB6awBdpUKD9jflb0SHzUv\nnKBds0pjBqAlkd25HN7rOrFleaJ1/
    →ctaJxQZBKT5ZPt0m9STJEadao0xAH0ahmbWn\nnOlFuhjuefXKNegV4We0+UXgVCwOPjdAvBbI+e0ocS3MFEVzG6uBQE3xDk3Sz
    →rOPnk3sgrDQoo//fb4hVC1CLQJ13hef4Y53CI\nnrU7m2Ys6xt0nUW7/
    →vGT1M0NPagMBAAGjQjBAMA4GA1UdDwEB/wQEAWIBBjAPBgNV\nnHRMBAf8EBTADAQH/
    →MB0GA1UdDgQWBBR5tFnme7b15AFzgAiIyBpY9umbbjANBgkq\nnhkiG9w0BAQsFAAOCAgEAVR9YqbyyqFDQDLHYGmkGjYkIrGF1
    →V91ZL\nnubhzEFnTIZd+50xx+7LSYK05gAvqFyFWhfQDlnrzuBZ6brJFe+GnY+EGPbk6ZGQ\nn3BebYhtF8GaV0nxvwuo77x/
    →Py9auJ/GpsMiu/X1+mvoiBOv/2X/qkSsisRcOj/
    →KK\nnNftY2PwByVS5UcBmiogziUwthDyC3+6WVwW6LLv3xLfHTjuCvJHIInNzktHCgKQ5\nnORAZI4JMPJ+GslWYHb4phowim57ia
    →+sKAIuvtd7u+Nxe5AW0wderL1N8NwdC\nnjNPElpzVmbUq4JUagEiuTDkHszxHpFKVK7q4+63SM1N95R1NbdWhscdCb+ZAJzVc\nno
    →1lvh+wjChP4kqK0J2qxq\nn4RgqsahDYVvTH9w7jXbyLeiNdd8XM2w9U/t7y0Fff/
    →9yi0GE44Za4rF2LN9d11TPA\nnmRGunUHBcnWEvgJBQ19nJEiU0Zsnvgc/
    →ubhPgXRR4Xq37Z0j4r7g1SgEEzwxA57d\nnemyPxcgYxn/eR44/KJ4EBs+lVDR3veyJm+kXQ99b21/
    →+jh5XosiAnX5iitreGCc=\n-----END CERTIFICATE-----"

```

(continues on next page)

(continued from previous page)

```

    },
    "wamp_onion_url": "",
    "wamp_url": ""
  },
  "electrum-mainnet": {
    "address_explorer_url": "https://blockstream.info/address/",
    "bech32_prefix": "bc",
    "bip21_prefix": "bitcoin",
    "blob_server_onion_url": "",
    "blob_server_url": "",
    "csv_buckets": [],
    "development": false,
    "electrum_onion_url": "explorerzydxu5ecjrkwcayqybizmpjjznk5izmitf2modhcsuqlid.
    onion:110",
    "electrum_tls": true,
    "electrum_url": "blockstream.info:700",
    "liquid": false,
    "mainnet": true,
    "max_reorg_blocks": 144,
    "name": "Bitcoin (Electrum)",
    "network": "electrum-mainnet",
    "p2pkh_version": 0,
    "p2sh_version": 5,
    "pin_server_onion_url": "http://
    mrrxtq6tjpbnbm7vh5jt6mpjctn7ggyfy5wegvbef3x7jrnqawlmid.onion",
    "pin_server_public_key":
    "0332b7b1348bde8ca4b46b9dcc30320e140ca26428160a27bdbfc30b34ec87c547",
    "pin_server_url": "https://jadepin.blockstream.com",
    "price_onion_url": "http://
    greenv32e5p4rax6dmfgeb4zzl7kq2fbmizd7miyava2actplmipyx2qd.onion/prices",
    "price_url": "https://green-bitcoin-mainnet.blockstream.com/prices",
    "server_type": "electrum",
    "service_chain_code": "",
    "service_pubkey": "",
    "spv_enabled": false,
    "spv_multi": false,
    "spv_servers": [],
    "tx_explorer_url": "https://blockstream.info/tx/",
    "wamp_cert_pins": [],
    "wamp_cert_roots": [
      "\n-----BEGIN CERTIFICATE-----\nMIICxjCCAk2gAwIBAgIRALO93/
    inhFu86QOgQTWzSkUwCgYIKoZIzj0EAwMwTzEL\nMAkGA1UEBhMCVVMxKTAnBgNVBAoTIEludGVybmV0IFNlY3VyaXR5IFJlc2V5
    xwLcA/TmC9N/A8AmzfwdAVXMpcuBe8qQyWj\n+240JxP2T35p0wKZXuskR5LBJJvmsSGPwSSB/
    GjMH2m6WPUZIVd0xhajggEIMIIB\nBDAOBgNVHQ8BAf8EBAMCAYYwHQYDVR0lBBYwFAYIKwYBBQUHAwIGCCsGAQUFBwMB\nnMBIIC
    wQIMAYBAf8CAQAwHQYDVR0OBBYEFFrz7Sv8NsI3eblSMOpUb89V\nnyy6sMB8GA1UdIwQYMBaAFHxClq7eS0g7+pL4nozPbYupc
    AwCd89eYhN1Lr3pD5xcSAQH8suzYHK019YST8pE9kLJ03uGqQIwWrGxtO3q\nnYJkgsTgDyJ2gJrjubilK9sZmHzOa25JK1fUpI
    Lgul/\n-----END CERTIFICATE-----",
      "\n-----BEGIN CERTIFICATE-----\nMIICxjCCAkygAwIBAgIQTtI99q9+x/
    mwXHJv+VEqdzAKBggqhkJOPQDazBPMQsw\nnCQYDVQQGEwJVUzEpMCcGA1UEChMgSW50ZXJuZXQgU2VjdXJpdHkgUmVzZWZyY2
    LbBVDxCsnsI7vuf4xp4Cu+ETNgGGE\nnyBqJ3j8iUwe5Yt/qfSrRf1/
    D5R58duaJ+IvLRXeASRqEL+VkdXrW3qOCAQgwggEE\nnMA4GA1UdDwEB/
    wQEAwIBhjAdBgNVHSUEFjAUBgggrBgEFBQcDAgYIKwYBBQUHAwEw\nnEgYDVR0TAQH/BAgwBgEB/
    wBADAdBgNVHQ4EFqQUbZkq9U0C6+MRWC6km+NPS7x\nn6kQwHwYDVR0lJBBgwFoAUfEKWrt5LSDv6kviejM9ti6lYn5UwMgYIKw
    No+KQ\nnUAM1u34E36neL/Zba7ombkIOchSgx1iVxzqtFWGddgoG+tpRPWhuhhn\n-----END
    CERTIFICATE-----",
      "\n-----BEGIN CERTIFICATE-----
    \nMIIFWjCCAOKgAwIBAgIQbkepxUtHDA3sM9CJuRz04TANBgkqhkiG9w0BAQwFADBH\nnMQswCQYDVQQGEwJVUzEiMCAgA1UEChMw
    vo27xqLpeXo4xL+Sv2sfnOhB2x+cWX3u+58qPpvBKJXgeqUqv4IyflpLGcY9vX\nnmX7wC1
    SCYe7\nnzUjwTcLCoikU7rPWRnWr4+wB7CeMfGCwcDfLqZtbBkOtdh+JhpFaz2weaSUKK0P\nnfylqAj+lug8aJRT7oM6iCsVl
    kB+Xj9e1x3+naH+uzfsQ55iVe0vSbvlgHR6xyKu44LtcXFilwR06zqkUsp\nnzBmkMiVOKvFiRNACzqrOSbTqn3yDsEB750Orp2y
    vJsfO3C12LO0nRC92wOIAK/
    1TD7Cn1TsNsYqiA94xrcx36m97PtbfkSIS5r762DL8EGMUUXLeXdYw\nnk70paDPvOmsB4om3xPXV2V4J95eSRQAogB/
    mqgghtqmxlbCluQ0WEdrHbEg8QOB+\nndVrNVjzrLw5y0vtOUucD/SVRNuJLDWcfr0wbrM7Rv1/
    oFB2ACYPrIrnqYNxgF\nnlQIDAQABO1wQDAOBgNVHQ8BAf8EBAMCAQYwDwYDVR0TAQH/BAUwAwEB/
    12BUN\nH41FE\nH5H2\n127\nH6\n128\nH7\n129\nH8\n130\nH9\n131\nH10\n132\nH11\n133\nH12\n134\nH13\n135\nH14\n136\nH15\n137\nH16\n138\nH19\n139\nH20\n140\nH21\n141\nH22\n142\nH23\n143\nH24\n144\nH25\n145\nH26\n146\nH27\n147\nH28\n148\nH29\n149\nH50\n151\nH52\n153\nH54\n155\nH56\n157\nH58\n159\nH60\n161\nH62\n163\nH64\n165\nH66\n167\nH68\n169\nH70\n171\nH72\n173\nH74\n175\nH76\n177\nH78\n179\nH80\n181\nH82\n183\nH84\n185\nH86\n187\nH88\n189\nH90\n191\nH92\n193\nH94\n195\nH96\n197\nH98\n199\nH200\n201\nH202\n203\nH204\n205\nH206\n207\nH208\n209\nH210\n211\nH212\n213\nH214\n215\nH216\n217\nH218\n219\nH220\n221\nH222\n223\nH224\n225\nH226\n227\nH228\n229\nH230\n231\nH232\n233\nH234\n235\nH236\n237\nH238\n239\nH240\n241\nH242\n243\nH244\n245\nH246\n247\nH248\n249\nH250\n251\nH252\n253\nH254\n255\nH256\n257\nH258\n259\nH260\n261\nH262\n263\nH264\n265\nH266\n267\nH268\n269\nH270\n271\nH272\n273\nH274\n275\nH276\n277\nH278\n279\nH280\n281\nH282\n283\nH284\n285\nH286\n287\nH288\n289\nH290\n291\nH292\n293\nH294\n295\nH296\n297\nH298\n299\nH300\n301\nH302\n303\nH304\n305\nH306\n307\nH308\n309\nH310\n311\nH312\n313\nH314\n315\nH316\n317\nH318\n319\nH320\n321\nH322\n323\nH324\n325\nH326\n327\nH328\n329\nH330\n331\nH332\n333\nH334\n335\nH336\n337\nH338\n339\nH340\n341\nH342\n343\nH344\n345\nH346\n347\nH348\n349\nH350\n351\nH352\n353\nH354\n355\nH356\n357\nH358\n359\nH360\n361\nH362\n363\nH364\n365\nH366\n367\nH368\n369\nH370\n371\nH372\n373\nH374\n375\nH376\n377\nH378\n379\nH380\n381\nH382\n383\nH384\n385\nH386\n387\nH388\n389\nH390\n391\nH392\n393\nH394\n395\nH396\n397\nH398\n399\nH400\n401\nH402\n403\nH404\n405\nH406\n407\nH408\n409\nH410\n411\nH412\n413\nH414\n415\nH416\n417\nH418\n419\nH420\n421\nH422\n423\nH424\n425\nH426\n427\nH428\n429\nH430\n431\nH432\n433\nH434\n435\nH436\n437\nH438\n439\nH440\n441\nH442\n443\nH444\n445\nH446\n447\nH448\n449\nH450\n451\nH452\n453\nH454\n455\nH456\n457\nH458\n459\nH460\n461\nH462\n463\nH464\n465\nH466\n467\nH468\n469\nH470\n471\nH472\n473\nH474\n475\nH476\n477\nH478\n479\nH480\n481\nH482\n483\nH484\n485\nH486\n487\nH488\n489\nH490\n491\nH492\n493\nH494\n495\nH496\n497\nH498\n499\nH500\n501\nH502\n503\nH504\n505\nH506\n507\nH508\n509\nH510\n511\nH512\n513\nH514\n515\nH516\n517\nH518\n519\nH520\n521\nH522\n523\nH524\n525\nH526\n527\nH528\n529\nH530\n531\nH532\n533\nH534\n535\nH536\n537\nH538\n539\nH540\n541\nH542\n543\nH544\n545\nH546\n547\nH548\n549\nH550\n551\nH552\n553\nH554\n555\nH556\n557\nH558\n559\nH560\n561\nH562\n563\nH564\n565\nH566\n567\nH568\n569\nH570\n571\nH572\n573\nH574\n575\nH576\n577\nH578\n579\nH580\n581\nH582\n583\nH584\n585\nH586\n587\nH588\n589\nH590\n591\nH592\n593\nH594\n595\nH596\n597\nH598\n599\nH600\n601\nH602\n603\nH604\n605\nH606\n607\nH608\n609\nH610\n611\nH612\n613\nH614\n615\nH616\n617\nH618\n619\nH620\n621\nH622\n623\nH624\n625\nH626\n627\nH628\n629\nH630\n631\nH632\n633\nH634\n635\nH636\n637\nH638\n639\nH640\n641\nH642\n643\nH644\n645\nH646\n647\nH648\n649\nH650\n651\nH652\n653\nH654\n655\nH656\n657\nH658\n659\nH660\n661\nH662\n663\nH664\n665\nH666\n667\nH668\n669\nH670\n671\nH672\n673\nH674\n675\nH676\n677\nH678\n679\nH680\n681\nH682\n683\nH684\n685\nH686\n687\nH688\n689\nH690\n691\nH692\n693\nH694\n695\nH696\n697\nH698\n699\nH700\n701\nH702\n703\nH704\n705\nH706\n707\nH708\n709\nH710\n711\nH712\n713\nH714\n715\nH716\n717\nH718\n719\nH720\n721\nH722\n723\nH724\n725\nH726\n727\nH728\n729\nH730\n731\nH732\n733\nH734\n735\nH736\n737\nH738\n739\nH740\n741\nH742\n743\nH744\n745\nH746\n747\nH748\n749\nH750\n751\nH752\n753\nH754\n755\nH756\n757\nH758\n759\nH760\n761\nH762\n763\nH764\n765\nH766\n767\nH768\n769\nH770\n771\nH772\n773\nH774\n775\nH776\n777\nH778\n779\nH780\n781\nH782\n783\nH784\n785\nH786\n787\nH788\n789\nH790\n791\nH792\n793\nH794\n795\nH796\n797\nH798\n799\nH800\n801\nH802\n803\nH804\n805\nH806\n807\nH808\n809\nH810\n811\nH812\n813\nH814\n815\nH816\n817\nH818\n819\nH820\n821\nH822\n823\nH824\n825\nH826\n827\nH828\n829\nH830\n831\nH832\n833\nH834\n835\nH836\n837\nH838\n839\nH840\n841\nH842\n843\nH844\n845\nH846\n847\nH848\n849\nH850\n851\nH852\n853\nH854\n855\nH856\n857\nH858\n859\nH860\n861\nH862\n863\nH864\n865\nH866\n867\nH868\n869\nH870\n871\nH872\n873\nH874\n875\nH876\n877\nH878\n879\nH880\n881\nH882\n883\nH884\n885\nH886\n887\nH888\n889\nH890\n891\nH892\n893\nH894\n895\nH896\n897\nH898\n899\nH900\n901\nH902\n903\nH904\n905\nH906\n907\nH908\n909\nH910\n911\nH912\n913\nH914\n915\nH916\n917\nH918\n919\nH920\n921\nH922\n923\nH924\n925\nH926\n927\nH928\n929\nH930\n931\nH932\n933\nH934\n935\nH936\n937\nH938\n939\nH940\n941\nH942\n943\nH944\n945\nH946\n947\nH948\n949\nH950\n951\nH952\n953\nH954\n955\nH956\n957\nH958\n959\nH960\n961\nH962\n963\nH964\n965\nH966\n967\nH968\n969\nH970\n971\nH972\n973\nH974\n975\nH976\n977\nH978\n979\nH980\n981\nH982\n983\nH984\n985\nH986\n987\nH988\n989\nH990\n991\nH992\n993\nH994\n995\nH996\n997\nH998\n999\nH1000\n1001\nH1002\n1003\nH1004\n1005\nH1006\n1007\nH1008\n1009\nH1010\n1011\nH1012\n1013\nH1014\n1015\nH1016\n1017\nH1018\n1019\nH1020\n1021\nH1022\n1023\nH1024\n1025\nH1026\n1027\nH1028\n1029\nH1030\n1031\nH1032\n1033\nH1034\n1035\nH1036\n1037\nH1038\n1039\nH1040\n1041\nH1042\n1043\nH1044\n1045\nH1046\n1047\nH1048\n1049\nH1050\n1051\nH1052\n1053\nH1054\n1055\nH1056\n1057\nH1058\n1059\nH1060\n1061\nH1062\n1063\nH1064\n1065\nH1066\n1067\nH1068\n1069\nH1070\n1071\nH1072\n1073\nH1074\n1075\nH1076\n1077\nH1078\n1079\nH1080\n1081\nH1082\n1083\nH1084\n1085\nH1086\n1087\nH1088\n1089\nH1090\n1091\nH1092\n1093\nH1094\n1095\nH1096\n1097\nH1098\n1099\nH1100\n1101\nH1102\n1103\nH1104\n1105\nH1106\n1107\nH1108\n1109\nH1110\n1111\nH1112\n1113\nH1114\n1115\nH1116\n1117\nH1118\n1119\nH1120\n1121\nH1122\n1123\nH1124\n1125\nH1126\n1127\nH1128\n1129\nH1130\n1131\nH1132\n1133\nH1134\n1135\nH1136\n1137\nH1138\n1139\nH1140\n1141\nH1142\n1143\nH1144\n1145\nH1146\n1147\nH1148\n1149\nH1150\n1151\nH1152\n1153\nH1154\n1155\nH1156\n1157\nH1158\n1159\nH1160\n1161\nH1162\n1163\nH1164\n1165\nH1166\n1167\nH1168\n1169\nH1170\n1171\nH1172\n1173\nH1174\n1175\nH1176\n1177\nH1178\n1179\nH1180\n1181\nH1182\n1183\nH1184\n1185\nH1186\n1187\nH1188\n1189\nH1190\n1191\nH1192\n1193\nH1194\n1195\nH1196\n1197\nH1198\n1199\nH1200\n1201\nH1202\n1203\nH1204\n1205\nH1206\n1207\nH1208\n1209\nH1210\n1211\nH1212\n1213\nH1214\n1215\nH1216\n1217\nH1218\n1219\nH1220\n1221\nH1222\n1223\nH1224\n1225\nH1226\n1227\nH1228\n1229\nH1230\n1231\nH1232\n1233\nH1234\n1235\nH1236\n1237\nH1238\n1239\nH1240\n1241\nH1242\n1243\nH1244\n1245\nH1246\n1247\nH1248\n1249\nH1250\n1251\nH1252\n1253\nH1254\n1255\nH1256\n1257\nH1258\n1259\nH1260\n1261\nH1262\n1263\nH1264\n1265\nH1266\n1267\nH1268\n1269\nH1270\n1271\nH1272\n1273\nH1274\n1275\nH1276\n1277\nH1278\n1279\nH1280\n1281\nH1282\n1283\nH1284\n1285\nH1286\n1287\nH1288\n1289\nH1290\n1291\nH1292\n1293\nH1294\n1295\nH1296\n1297\nH1298\n1299\nH1300\n1301\nH1302\n1303\nH1304\n1305\nH1306\n1307\nH1308\n1309\nH1310\n1311\nH1312\n1313\nH1314\n1315\nH1316\n1317\nH1318\n1319\nH1320\n1321\nH1322\n1323\nH1324\n1325\nH1326\n1327\nH1328\n1329\nH1330\n1331\nH1332\n1333\nH1334\n1335\nH1336\n1337\nH1338\n1339\nH1340\n1341\nH1342\n1343\nH1344\n1345\nH1346\n1347\nH1348\n1349\nH1350\n1351\nH1352\n1353\nH1354\n1355\nH1356\n1357\nH1358\n1359\nH1360\n1361\nH1362\n1363\nH1364\n1365\nH1366\n1367\nH1368\n1369\nH1370\n1371\nH1372\n1373\nH1374\n1375\nH1376\n1377\nH1378\n1379\nH1380\n1381\nH1382\n1383\nH1384\n1385\nH1386\n1387\nH1388\n1389\nH1390\n1391\nH1392\n1393\nH1394\n1395\nH1396\n1397\nH1398\n1399\nH1400\n1401\nH1402\n1403\nH1404\n1405\nH1406\n1407\nH1408\n1409\nH1410\n1411\nH1412\n1413\nH1414\n1415\nH1416\n1417\nH1418\n1419\nH1420\n1421\nH1422\n1423\nH1424\n1425\nH1426\n1427\nH1428\n1429\nH1430\n1431\nH1432\n1433\nH1434\n1435\nH1436\n1437\nH1438\n1439\nH1440\n1441\nH1442\n1443\nH1444\n1445\nH1446\n1447\nH1448\n1449\nH1450\n1451\nH1452\n1453\nH1454\n1455\nH1456\n1457\nH1458\n1459\nH1460\n1461\nH1462\n1463\nH1464\n1465\nH1466\n1467\nH1468\n1469\nH1470\n1471\nH1472\n1473\nH1474\n1475\nH1476\n1477\nH1478\n1479\nH1480\n1481\nH1482\n1483\nH1484\n1485\nH1486\n1487\nH1488\n1489\nH1490\n1491\nH1492\n1493\nH1494\n1495\nH1496\n1497\nH1498\n1499\nH1500\n1501\nH1502\n1503\nH1504\n1505\nH1506\n1507\nH1508\n1509\nH1510\n1511\nH1512\n1513\nH1514\n1515\nH1516\n1517\nH1518\n1519\nH1520\n1521\nH1522\n1523\nH1524\n1525\nH1526\n1527\nH1528\n1529\nH1530\n1531\nH1532\n1533\nH1534\n1535\nH1536\n1537\nH1538\n1539\nH1540\n1541\nH1542\n1543\nH1544\n1545\nH1546\n1547\nH1548\n1549\nH1550\n1551\nH1552\n1553\nH1554\n1555\nH1556\n1557\nH1558\n1559\nH1560\n1561\nH1562\n1563\nH1564\n1565\nH1566\n1567\nH1568\n1569\nH1570\n1571\nH1572\n1573\nH1574\n1575\nH1576\n1577\nH1578\n1579\nH1580\n1581\nH1582\n1583\nH1584\n1585\nH1586\n1587\nH1588\n1589\nH1590\n1591\nH1592\n1593\nH1594\n1595\nH1596\n1597\nH1598\n1599\nH1600\n1601\nH1602\n1603\nH1604\n1605\nH1606\n1607\nH1608\n1609\nH1610\n1611\nH1612\n1613\nH1614\n1615\nH1616\n1617\nH1618\n1619\nH1620\n1621\nH1622\n1623\nH1624\n1625\nH1626\n1627\nH1628\n1629\nH1630\n1631\nH1632\n1633\nH1634\n1635\nH1636\n1637\nH1638\n1639\nH1640\n1641\nH1642\n1643\nH1644\n1645\nH1646\n1647\nH1648\n1649\nH1650\n1651\nH1652\n1653\nH1654\n1655\nH1656\n1657\nH1658\n1659\nH1660\n1661\nH1662\n1663\nH1664\n1665\nH1666\n1667\nH1668\n1669\nH1670\n1671\nH1672\n1673\nH1674\n1675\nH1676\n1677\nH1678\n1679\nH1680\n1681\nH1682\n1683\nH1684\n1685\nH1686\n1687\nH1688\n1689\nH1690\n1691\nH1692\n1693\nH1694\n1695\nH1696\n1697\nH1698\n1699\nH1700\n1701\nH1702\n1703\nH1704\n1705\nH1706\n1707\nH1708\n1709\nH1710\n1711\nH1712\n1713\nH1714\n1715\nH1716\n1717\nH1718\n1719\nH1720\n1721\nH1722\n1723\nH1724\n1725\nH1726\n1727\nH1728\n1729\nH1730\n1731\nH1732\n1733\nH1734\n1735\nH1736\n1737\nH1738\n1739\nH1740\n1741\nH1742\n1743\nH1744\n1745\nH1746\n1747\nH1748\n1749\nH1750\n1751\nH1752\n1753\nH1754\n1755\nH1756\n1757\nH1758\n1759\nH1760\n1761\nH1762\n1763\nH1764\n1765\nH1766\n1767\nH1768\n1769\nH1770\n1771\nH1772\n1773\nH1774\n1775\nH1776\n1777\nH1778\n1779\nH1780\n1781\nH1782\n1783\nH1784\n1785\nH1786\n1787\nH1788\n1789\nH1790\n1791\nH1792\n1793\nH1794\n1795\nH1796\n1797\nH1798\n1799\nH1800\n1801\nH1802\n1803\nH1804\n1805\nH1806\n1807\nH1808\n1809\nH1810\n1811\nH1812\n1813\nH1814\n1815\nH1816\n1817\nH1818\n1819\nH1820\n1821\nH1822\n1823\nH1824\n1825\nH1826\n1827\nH1828\n1829\nH1830\n1831\nH1832\n1833\nH1834\n1835\nH1836\n1837\nH1838\n1839\nH1840\n1841\nH1842\n1843\nH1844\n1845\nH1846\n1847\nH1848\n1849\nH1850\n1851\nH1852\n1853\nH1854\n1855\nH1856\n1857\nH1858\n1859\nH1860\n1861\nH1862\n1863\nH1864\n1865\nH1866\n1867\nH1868\n1869\nH1870\n1871\nH1872\n1873\nH1874\n1875\nH1876\n1877\nH1878\n1879\nH1880\n1881\nH1882\n1883\nH1884\n1885\nH1886\n1887\nH1888\n1889\nH1890\n1891\nH1892\n1893\nH1894\n1895\nH1896\n1897\nH1898\n1899\nH1900\n1901\nH1902\n1903\nH1904\n1905\nH1906\n1907\nH1908\n1909\nH1910\n1911\nH1912\n1913\nH1914\n1915\nH1916\n1917\nH1918\n1919\nH1920\n1921\nH1922\n1923\nH1924\n1925\nH1926\n1927\nH1928\n1929\nH1930\n1931\nH1932\n1933\nH1934\n1935\nH1936\n1937\nH1938\n1939\nH1940\n1941\nH1942\n1943\nH1944\n1945\nH1946\n1947\nH1948\n1949\nH1950\n1951\nH1952\n1953\nH1954\n1955\nH1956\n1957\nH1958\n1959\nH1960\n1961\nH1962\n1963\nH1964\n1965\nH1966\n1967\nH1968\n1969\nH1970\n1971\nH1972\n1973\nH1974\n1975\nH1976\n1977\nH1978\n1979\nH1980\n1981\nH1982\n1983\nH1984\n1985\nH1986\n1987\nH1988\n1989\nH1990\n1991\nH1992\n1993\nH1994\n1995\nH1996\n1997\nH1998\n1999\nH2000\n2001\nH2002\n2003\nH2004\n2005\nH2006\n2007\nH2008\n2009\nH2010\n2011\nH2012\n2013\nH2014\n2015\nH2016\n2017\nH2018\n2019\nH2020\n2021\nH2022\n2023\nH2024\n2025\nH2026\n2027\nH2028\n2029\nH2030\n2031\nH2032\n2033\nH2034\n2035\nH2036\n2037\nH2038\n2039\nH2040\n2041\nH2042\n2043\nH2044\n2045\nH2046\n2047\nH2048\n2049\nH2050\n2051\nH2052\n2053\nH2054\n2055\nH2056\n2057\nH2058\n2059\nH2060\n2061\nH2062\n2063\nH2064\n2065\nH2066\n2067\nH2068\n2069\nH2070\n2071\nH2072\n2073\nH2074\n2075\nH2076\n2077\nH2078\n2079\nH2080\n2081\nH2082\n2083\nH2084\n2085\nH2086\n2087\nH2088\n2089\nH2090\n2091\nH2092\n2093\nH2094\n2095\nH2096\n2097\nH2098\n2099\nH2100\n2101\nH2102\n2103\nH2104\n2105\nH2106\n2107\nH2108\n2109\nH2110\n2111\nH2112\n2113\nH2114\n2115\nH2116\n2117\nH2118\n2119\nH2120\n2121\nH2122\n2123\nH2124\n2125\nH2126\n2127\nH2128\n2129\nH2130\n2131\nH2132\n2133\nH2134\n2135\nH2136\n2137\nH2138\n2139\nH2140\n2141\nH2142\n2143\nH2144\n2145\nH2146\n2147\nH2148\n2149\nH2150\n2151\nH2152\n2153\nH2154\n2155\nH2156\n2157\nH2158\n2159\nH2160\n2161\nH2162\n2163\nH2164\n2165\nH2166\n2167\nH2168\n2169\nH2170\n2171\nH2172\n2173\nH2174\n2175\nH2176\n2177\nH2178\n2179
```

(continued from previous page)

```

"\n-----BEGIN CERTIFICATE-----
↪\nMIIFazCCA1OgAwIBAgIRAIQz7DSQONZRGPGu2OCiwAwDQYJKoZIhvcNAQELBQAw\nTzELMAkGA1UEBhMCVVMxKTAnBgNVBA
↪vVqbnvYATyjb3miGbESTtrFj/RQSa78f0uoxmyF+\n0TM8ukj13Xnfs7j/
↪EvEhmkvBioZxaUpmZmyPfjxwv60pIgbz5MDmgK7iS4+3mX6U\nnA5/
↪TR5d8mUgjU+g4rk8Kb4Mu0U1XjIB0ttov0DiNewNwIRt18jA8+o+u3dpjq+sW\nnT8KOEUt+zwvo/
↪7V3LvSye0rgTBIldHCNAymg4VMk7BPZ7hm/
↪ELNKjD+Jo2FR3qyH\nnB5T0Y3HsLuJvW5iB4Y1cNHlsdu87kGJ55tukmi8mxdAQ4Q7e2RCOFvu396j3x+UC\nnB5iPNgiV5+I31g
↪1JBdiB3QW0KtZB6awBdpUKD9jflb0SHzUv\nnKBds0pjBqAlkd25HN7rOrFleaJ1/
↪ctaJxQZBKT5ZPt0m9STJEadao0xAH0ahmbWn\nnOlFuhjuefXKnEgV4We0+UXgVCwOPjdAvBbI+e0ocS3MFEVzG6uBQE3xDk3Sz
↪rOPNk3sgrDQoo//fb4hVC1CLQJ13hef4Y53CI\nnrU7m2Ys6xt0nUW7/
↪vGT1M0NPAgMBAAGjQjBAMA4GA1UdDwEB/wQEAWIBBjAPBgNV\nnHRMBAf8EBTADAQH/
↪MB0GA1UdDgQWBRR5tFnme7b15AFzgAiIyBpY9umbbjANBgkq\nnhkiG9w0BAQsFAAOCAGEAVR9YqbyyqFDQDLHYGmkGJykIrGF1
↪V91ZL\nnubhzEFnTIZd+50xx+7LSYK05qAvqFyFWhfQDlnrzuBZ6brJFe+GnY+EGPbk6ZGQ\nn3BebYhtF8GaV0nxvwuo77x/
↪Py9auJ/GpsMiu/X1+mvoiBOv/2X/qkSsisRcOj/
↪KK\nnNftY2PwByVS5UcBmiogziUwthDyC3+6WVwW6LLv3xLfHTjuCvjiHIInNzktHCgKQ5\nnORAZI4JMPJ+GslWYHb4phowim57ia
↪+sKAiuvtd7u+Nxe5AW0wdeR1N8NwdC\nnjNPElpzVmbUq4JUagEiuTDkHszsHpFKVK7q4+63SM1N95R1NbdWhscdCb+ZAJzVc\nno
↪1lvh+wjChP4kqK0J2qxq\nn4RgqsahDYVvTH9w7jXbyLeiNdd8XM2w9U/t7y0Ff/
↪9yi0GE44Za4rF2LN9d11TPA\nnmRGunUHBcnWEvgJBQ19nJEiU0Zsnvgc/
↪ubhPgXRR4Xq37Z0j4r7g1SgEEzwxAS7d\nnemyPxgcYxn/eR44/KJ4EBs+lVDR3veyJm+kXQ99b21/
↪+jh5Xos1AnX5iItreGCc=\n-----END CERTIFICATE-----"
    ],
    "wamp_onion_url": "",
    "wamp_url": ""
  },
  "electrum-testnet": {
    "address_explorer_url": "https://blockstream.info/testnet/address/",
    "bech32_prefix": "tb",
    "bip21_prefix": "bitcoin",
    "blob_server_onion_url": "",
    "blob_server_url": "",
    "csv_buckets": [],
    "development": false,
    "electrum_onion_url": "explorerzydxu5ecjrkwcayqybizmpjjznk5izmitf2modhcsuqlid.
↪onion:143",
    "electrum_tls": true,
    "electrum_url": "blockstream.info:993",
    "liquid": false,
    "mainnet": false,
    "max_reorg_blocks": 1008,
    "name": "Testnet (Electrum)",
    "network": "electrum-testnet",
    "p2pkh_version": 111,
    "p2sh_version": 196,
    "pin_server_onion_url": "http://
↪mrrxtq6tjpbnbm7vh5jt6mpjctn7ggyfy5wegvbeff3x7jrznqawlmid.onion",
    "pin_server_public_key":
↪"0332b7b1348bde8ca4b46b9dcc30320e140ca26428160a27bdbfc30b34ec87c547",
    "pin_server_url": "https://jadepin.blockstream.com",
    "price_onion_url": "http://
↪greent5yfxruca52pkqjtg02qdxijscqlastnv3jwzpmavvffldldm2yd.onion/prices",
    "price_url": "https://green-bitcoin-testnet.blockstream.com/prices",
    "server_type": "electrum",
    "service_chain_code": "",
    "service_pubkey": "",
    "spv_enabled": false,
    "spv_multi": false,
    "spv_servers": [],

```

(continues on next page)

(continued from previous page)

```

"tx_explorer_url": "https://blockstream.info/testnet/tx/",
"wamp_cert_pins": [],
"wamp_cert_roots": [
    "\n-----BEGIN CERTIFICATE-----\nMIICxjCCAk2gAwIBAgIRAL093/
    →inhFu86Q0gQTWzSkUwCgYIKoZIzj0EAwMwTzEL\nMAKGA1UEBhMCVVMxKTAnBgNVBAoTIEludGVybmV0IFNlY3VyaXR5IFJlc2
    →xwLCA/TmC9N/A8AmzfwdAVXMpcuBe8qQyWj\n+240JxP2T35p0wKZXusKR5LBjJvmsSGPwSSB/
    →GjMH2m6WPUZiVd0xhaJggEIMIIB\nBDAOBgNVHQ8BAf8EBAMCAYYhQYDVR01BBYwFAYIKwYBBQUHAWIGCCsGAQUFBwMB\nnMBI
    →wQIMAYBAf8CAQAwHQYDVR0OBBYEFFrz7Sv8NsI3ebLSMOpUb89V\nnyy6sMB8GA1UdIwQYMBaAFHxC1q7eS0g7+pL4nozPbYupc
    →AwCd89eYhNlXLR3pD5xcSAQH8suzYHKOl9YST8pE9kLJ03uGqQIwWrGxtO3q\nyYJkgsTgDyj2gJrjubilK9sZnHzOa25JK1fUpl
    →Lgul/\n-----END CERTIFICATE-----",
    "\n-----BEGIN CERTIFICATE-----\nMIICxjCCAkYgAwIBAgIQTtI99q9+x/
    →mwXhJjv+VEqdzAKBggqhkJOPQDzBPMQsw\nnCQYDVQQGEwJVUzEpMCCGA1UEChMgSW50ZXJuZXQgU2VjdXJpdHkgUmVzZW
    →LbBVDxCsNsI7vuf4xp4Cu+ETNggGE\nnyBqJ3j8iUwe5Yt/qfSrRf1/
    →D5R58duaJ+IvLRXeASRqEL+VkDXrW3qOCAQgwggEE\nnMA4GA1UdDwEB/
    →wQEAwIBhjAdBgNVHSUEFjAUBgggBgEFGQcDAgYIKwYBBQUHAWew\nnEgYDVR0TAQH/BAgwBgEB/
    →wIBADAdBgNVHQ4EFgQUUbZkq9U0C6+MRwWC6km+NPS7x\nn6kQwHwYDVR0jBBgwFoAUfEKWrt5LSDv6kviejM9ti6lyN5UwMgYIKw
    →No+KQ\nnUAMlu34E36neL/Zba7ombkIOchSgx1iVxztqFWGddgoG+tpPRPWhuhhn\nn-----END
    →CERTIFICATE-----",
    "\n-----BEGIN CERTIFICATE-----
    →\nMIIFWjCCAOKgAwIBAgIQbkepxUtHDA3sM9CJuRz04TANBgkqhkiG9w0BAQwFADBH\nnMQswCQYDVQQGEwJVUzEiMCAgA1UEChM
    →vo27xqLpeXo4xL+Sv2sfnohB2x+cWX3u+58qPpvBKJXqeqUqv4IyflPLGcY9vX\nnmX7wC17raKb0xlpHdu0QM+NOsROjyBhsS+
    →SCYe7\nnzUjwTcLCEOiKu7rPWRnWr4+wB7CEmFGCwCdfLqZtbBkOtdh+JhpFAz2weaSUKKOP\nnfyblqAj+lug8aJRT7oM6iCsVlq
    →kB+X+j9elx3+naH+uzfsQ551Ve0vSbvlgHR6xYku44LtcXFilWr06zqkUsp\nnzBmkMiVOKvFlRNACzqrOsBtqn3yDsEB7500rp2y
    →VjsPOS+C12LOO\nnRc92wO1AK/
    →1TD7Cn1TsNsYqiA94xrcx36m97PtbfkSIS5r762DL8EGMUUXLeXdyW\nnk70paDPvOmsB4om3xPXV2V4J95eSRQAogB/
    →mqgghtqmxlbCluQ0WEdrHbEg8QOB+\nndVrNVjzrLwW5y0vtOUucD/SVRNuJLDWcfr0wbrM7Rv1/
    →oFB2ACYPTTrIrnqYNxgF\nnlQIDAQABO0IwQDAOBgNVHQ8BAf8EBAMCAQYwDwYDVR0TAQH/BAUwAwEB/
    →zAdBgNV\nnHQ4EFgQU5K8rJnEaK0gnhS9SZizv8IkTcT4wDQYJKoZIhvcNAQEMBQADggIBADiW\nnCu49tJYeX++dnAsznyvgvy3S
    →UEBZvXw6Z\nnXPYfcX3v73svfu021pdwCxXu1xWaJ0140k4DLh9+42FpLFZXvRq4d2h9mREruZR\nngyFmxhE+885H7pwoHyXa/
    →6xmld01D1zvICxi/ZG6qcz8WpyTgYmpl0p8WnK0OdC3\nnd8t5/
    →Wk6kjtftbjhlRn7pYL15iJdfOBL07q9bgsiGleGZbYwE8na6SfZu6W0eX6Dv\nnJ4J2QPim01hcDyxC2kLGe4g0x8HYRZvBPsVhH
    →Zg\nnDdIx02OYI5NaAIFitO/
    →Nis3Jz5nu2Z6qNuFoS3FJFDYoOj0dzpqPJeaAcWErtXvM\nn+SUWgeExX6GjfhaknBZqlxi9dnKlC54dNuYvoS++cJEPqOba+MS
    →ZRBils6Kl0obsXs7X9\nnSQ98POyDGCBDTtWTurQ0sR8WNh8M5mQ5Fkzc4P4dyKliPUDqysU0ArSuiYgzNdwS\nnE3PYJ/
    →HQcu51OyLemGhmW/HGY0dVHLq1CFF1pkgl\nn-----END CERTIFICATE-----",
    "\n-----BEGIN CERTIFICATE-----
    →\nMIIFazCCA1OgAwIBAgIRAIQz7DSQONZRGpGu2OCiAwAdDQYJKoZIhvcNAQELBQAw\nnTzELMAKGA1UEBhMCVVMxKTAnBgNVBAoTIEludGVybmV0IFNlY3VyaXR5IFJlc2
    →vVqbvYATyjb3miGbESTtrFj/RQSa78f0uoxmyF+\nn0TM8ukj13Xnfs7j/
    →EvEhmkvBioZxaUpmZmyPfjxwv60pIgbz5MDmgK7iS4+3mX6U\nnA5/
    →TR5d8mUgjU+g4rk8Kb4Mu0U1XjIB0ttov0DiNewNwIRt18jA8+o+u3dpjq+sW\nnT8KOEut+zwvo/
    →7V3LvSye0rgTBIldHCNAymg4VMk7BPZ7hm/
    →ELNKjD+Jo2FR3qyH\nnB5T0Y3HsLuJvW5iB4Y1cNHlsdu87kGJ55tukmi8mxdAQ4Q7e2RCOFvu396j3x+UC\nnB5iPNgiV5+I3lgo
    →1JBdiB3QW0ktZB6awBdpUKD9jflb0ShzUv\nnKBds0pjBqAlkd25HN7rOrFleaJ1/
    →ctaJxQZBKT5ZPt0m9STJEadao0xAH0ahmbWn\nnOlFuhjuefXKnEgV4We0+UXgVCwOPjdAvBbI+e0ocS3MFEvzG6uBQE3xDk3Szy
    →rOPNk3sgrDQoo//fb4hVC1CLQJ13hef4Y53CI\nnrU7m2Ys6xtOnUW7/
    →vGT1M0NPAgMBAAGjQjBAMA4GA1UdDwEB/wQEAwIBBjAPBgNV\nnHRMBAf8EBTADAQH/
    →MB0GA1UdDgQWBBR5tFnme7b15AFzgAiIyBpY9umbbjANBgkq\nnhkiG9w0BAQsFAAOCAgEAVR9YqbyyqFDQDLHYGmkGjYkIrGF12
    →V91ZL\nnubhzEFnTIZd+50xx+7LSYK05qAvqFyFWhfQDlnrzuBZ6brJFe+GnY+EGPbk6ZGQ\nn3BebYhtF8GaV0nxvwuo77x/
    →Py9auJ/GpsMiu/X1+mvoiBOv/2X/qkSsisRcOj/
    →KK\nnNftY2PwByVS5uCbMiogziUwthDyC3+6WVwW6LLv3xLfHTjuCvJHIInNzktHCgKQ5\nnORAZI4JMPJ+GslWYHb4phowim57ia
    →+sKAiuvtd7u+Nxe5AW0wdeRlN8NwdC\nnjNPElpzVmbUq4JUagEiuTDkHszsHpFKVK7q4+63SM1N95R1NbdWhscdCb+ZAJzVc\nno
    →1lvh+wjChP4kqKOJ2qxq\nn4RgqsahDYVvTH9w7jXbyLeiNdd8XM2w9U/t7y0Ff/
    →9yi0GE44Za4rF2LN9d11TPA\nnmRGunUHBcnWEvgJBQ19nJEiU0Zsnvgc/
    →ubhPgXRR4Xq37Z0j4r7g1SgEEzwxA57d\nnemyPxgcYxn/eR44/KJ4EBs+lvDR3veyJm+kXQ99b21/
    →+jh5Xos1AnX5iItreGCC=\nn-----END CERTIFICATE-----"
],
"wamp_onion_url": "",

```

(continues on next page)

(continued from previous page)

```

    "wamp_url": ""
  },
  "electrum-testnet-liquid": {
    "address_explorer_url": "https://blockstream.info/liquidtestnet/address/",
    "asset_registry_onion_url": "http://
↳ lhquhzzpzg5tyymcqep24fynpzzqgg3m3rlh7ascnw5cpqsro35bfxyd.onion/testnet/",
    "asset_registry_url": "https://assets-testnet.blockstream.info/",
    "bech32_prefix": "tex",
    "bip21_prefix": "liquidtestnet",
    "blech32_prefix": "tlq",
    "blinded_prefix": 23,
    "blob_server_onion_url": "",
    "blob_server_url": "",
    "csv_buckets": [],
    "development": false,
    "electrum_onion_url": "explorerzydxu5ecjrkwcayqybizmpjjznk5izmitf2modhcsuqlid.
↳ onion:587",
    "electrum_tls": true,
    "electrum_url": "blockstream.info:465",
    "liquid": true,
    "mainnet": false,
    "max_reorg_blocks": 2,
    "name": "Testnet Liquid (Electrum)",
    "network": "electrum-testnet-liquid",
    "p2pkh_version": 36,
    "p2sh_version": 19,
    "pin_server_onion_url": "http://
↳ mrrxtq6tjpbnbm7vh5jt6mpjctn7ggyfy5wegvbef3x7jrznqawlmid.onion",
    "pin_server_public_key":
↳ "0332b7b1348bde8ca4b46b9dcc30320e140ca26428160a27bdbfc30b34ec87c547",
    "pin_server_url": "https://jadepin.blockstream.com",
    "policy_asset": "144c654344aa716d6f3abcc1ca90e5641e4e2a7f633bc09fe3baf64585819a49
↳ ",
    "price_onion_url": "http://
↳ greent5yfxruca52pkqjtgo2qdxijscqlastnv3jwzpmavvffdlm2yd.onion/prices",
    "price_url": "https://green-bitcoin-testnet.blockstream.com/prices",
    "server_type": "electrum",
    "service_chain_code": "",
    "service_pubkey": "",
    "spv_enabled": false,
    "spv_multi": false,
    "spv_servers": [],
    "tx_explorer_url": "https://blockstream.info/liquidtestnet/tx/",
    "wamp_cert_pins": [],
    "wamp_cert_roots": [
      "\n-----BEGIN CERTIFICATE-----\nMIICXjCCAk2gAwIBAgIRAL093/
↳ inhFu86QOgQTWzSkUwCgYIKoZIzj0EAwMwTzEL\nMAkGA1UEBhMCVVMxKTAnBgNVBAoTIEludGVybmV0IFNlY3VyaXR5IFJlc2
↳ xwLCA/TmC9N/A8AmzfwdAVXMpcuBe8qQyWj\n+240JxP2T35p0wKZXusKR5LBJJvmsSGPwSSB/
↳ GjMH2m6WPUZlvd0xhajggEIMIIB\nBDAOBgNVHQ8BAf8EBAMCAYYwHQYDVR01BBYwFAYIKwYBBQUHAWIGCCSGAQUFBwMB\nnMBI
↳ wQIMAYBAf8CAQAwHQYDVR0OBBYEFFrz7Sv8NsI3ebLSMOpUb89V\nnyy6sMB8GA1UdIwQYMBaAFHxC1q7eS0g7+pL4nozPbYupc
↳ AwCd89eYhN1XLr3pD5xcSAQh8suzYHK019YST8pE9kLJ03uGqQIwWrGxtO3q\nnYJkgsTgDy2j2JrjubilK9sZnHzOa25JK1fUp
↳ Lgu1/\n-----END CERTIFICATE-----",
      "\n-----BEGIN CERTIFICATE-----\nMIICXjCCAkygAwIBAgIQTtI99q9+x/
↳ mwXHJv+VEqdzAKBggqhkJOPQDAzBPMQsw\nnCQYDVQQGEwJVUzEpMCcGA1UEChMgSW50ZXJuZXQgU2VjdXJpdHkgUmVzZWYy
↳ LbBVDxCsNsI7vuf4xp4Cu+ETNggGE\nnyBqJ3j8iUwe5Yt/qfSrRf1/
↳ D5R58duaJ+IvLRXeASRqEL+VkdXrW3qOCAQgwgwGEE\nnMA4GA1UdDwEB/
↳ wQEAWIBhjadBgNVHSUEFjAUBgggrBgEFBQcDAGYIKwYBBQUHAWew\nnEgYDVR0TAQH/BAgwBgEB/
↳ wIBADAdBgNVHQ4EFgQUbZkq9U0C6+MRwWC6km+NPS7x\nn6kQwHwYDVR01jBBgwFoAUfEKWr
↳ No+KQ\nnUAM1u34E36neL/Zba7ombkIOchSgx1iVxztqFWGddgoG+tpRPWhuhhn\nn-----END
↳ CERTIFICATE-----",

```

(continues on next page)

(continued from previous page)

```

"\n-----BEGIN CERTIFICATE-----
→ \nMIIFWjCCA0KgAwIBAgIQbkepxUtHDA3sM9CJuRz04TANBgqhkiG9w0BAQwFADBH\nMQswCQYDVQQGEwJVUzEiMCAGA1UEChM=
→ vo27xqLpeXo4xL+Sv2sfnohB2x+cWX3u+58qPpvBKJXqeqUqv4IyflpLGcY9vX\nnmX7wCl7raKb0xlpHdU0QM+NOsROjyBhsS+z
→ SCYe7\nzUjwTcLceoiKu7rPWRnWr4+wB7CeMfGCwcdFlqZtbBkOtdh+JhpFAz2weaSUKK0P\nnfyblqAj+lug8aJRT7oM6iCsVlq
→ kB+Xj9e1x3+naH+uzfsQ551Ve0vSbvlGHR6xYKu44LtcXFilWr06zqkUsp\nnzBmkMiVOKvFlRNACzqrOSbtqn3yDsEB7500rp2
→ VjsPOS+C12L00\nRc92w01AK/
→ 1TD7Cn1TsNsYqiA94xrcx36m97PtbfkSIS5r762DL8EGMUUXLeXdYW\nnk70paDPvOmbS4om3xPXV2V4J95eSRQAogB/
→ mgghtqmxlbCluQ0WEdrHbEg8QOB+\nndVrNVjzRlwW5y0vtOUucxD/SVRNuJLDWcfr0wbrM7Rv1/
→ oFB2ACYPTIrinqYNxgF\nnlQIDAQABO0IwQDAOBgNVHQ8BAf8EBAMCAQYwDwYDVR0TAQH/BAUwAwEB/
→ zAdBgNV\nnHQ4EFgQU5K8rJnEaK0gnhS9SZizv8IkTcT4wDQYJKoZIhvcNAQEMBAQDggIBADiW\nnCu49tJYeX++dnAsznyvgv3S
→ UEBZvXw6Z\nnXPYfcX3v73svfu021pdwCxCXu11xWaj0140k4DLh9+42FpLFZxvRq4d2h9mREruZR\nngyFmxhE+885H7pwoHyXa/
→ 6xmld01D1zvICxi/ZG6qcz8WpyTgYmpl0p8WnK0OdC3\nnd8t5/
→ Wk6kjftbjhlRn7pYL15iJdfOBL07q9bgsiGleGZbYwE8na6SfZu6W0eX6Dv\nnJ4J2QPim01hcDyx2kLGe4g0x8HYRZvBPsVhH
→ Zg\nnDdIx02OYI5NaAIFItO/
→ Nis3Jz5nu2Z6qNuFoS3FJFDYoOj0dzpqPJeaAcWErtXvM\nn+SUWgeExX6GjfhaknBZqlxi9dnKlC54dNuYvoS++cJEPqOba+MS
→ ZRBils6Kl0obsXs7X9\nnSQ98POyDGCBDTtWTurQ0sR8WNh8M5mQ5Fkzc4P4dyKliPUDqysU0ArSuiYgzNdws\nnE3PYJ/
→ HQcu51OyLemGhmW/HGY0dVHLq1CFF1pklgl\nn-----END CERTIFICATE-----",

"\n-----BEGIN CERTIFICATE-----
→ \nMIIFazCCA1OgAwIBAgIRAIQz7DSQONZRGPGu2OCiwAwDQYJKoZIhvcNAQELBQAw\nnTzELMAkGA1UEBhMCVVMxKTAnBgNVBAQ=
→ vVqbvYATyjb3miGbESTtrFj/RQSa78f0uoxmyF+\nnoTM8ukj13Xnfs7j/
→ EvEhmkvBioZxaUpmZmyPfjxwv60pIgbz5MDmgK7iS4+3mX6U\nnA5/
→ TR5d8mUgjU+g4rk8Kb4Mu0U1XjIB0ttov0DiNewNwIRt18jA8+o+u3dpjq+sW\nnT8KOEut+zwvo/
→ 7V3LvSye0rgTBILDHcNAymg4VMk7BPZ7hm/
→ ELNKjD+Jo2FR3qyH\nnB5T0Y3HsLuJvW5iB4Y1cNHlsdu87kGJ55tukmi8mxdAQ4Q7e2RCOFvu396j3x+UC\nnB5iPNgiV5+I31g/
→ lJBdiB3QW0KtZB6awBdpUKD9jflb0SHzUv\nnKBds0pjBqAlkd25HN7rOrFleaJ1/
→ ctaJxQZBKT5ZPt0m9STJEadao0xAH0ahmbWn\nnOlFuhjuefXKnEgV4We0+UXgVCwOPjdAvBbI+e0ocS3MFEVzG6uBQE3xDk3Szy
→ rOPNk3sgrDQoo//fb4hVC1CLQJ13hef4Y53CI\nnrU7m2Ys6xt0nUW7/
→ vGT1M0NPagMBAAGjQjBAMA4GA1UdDwEB/wQEAWIBBjAPBgNV\nnHRMBAf8EBTADAQH/
→ MB0GA1UdDgQWBBR5tFnme7b15AFzgAiIyBpY9umbbjANBgkq\nnhkiG9w0BAQwFAAOCAgEAVR9YqbyyqFDQDLHYGmkGjYkIrGF1
→ V91ZL\nnubhzEFnTIZd+50xx+7LSYK05qAvqFyFWhfQDlnrzuBZ6brJFe+GnY+EGPbk6ZGQ\nn3BebYhtF8GaV0nxvwuo77x/
→ Py9auJ/GpsMiu/X1+mvoiBOv/2X/qkSsisRcOj/
→ KK\nnNftY2PwByVS5UcBmiogziUwthDyC3+6WVwW6LLv3xLfHTjuCvJHIInNzktHCgKQ5\nnORAZI4JMPJ+GslWYHb4phowim57ia
→ +sKAIuvtd7u+Nxe5AW0wdeRlN8NwdC\nnjNPelPzVmbUq4JUagEiuTDkHszxHpFKVK7q4+63SM1N95R1NbdWhscdCb+ZAJzVc\nno
→ 1lvh+wjChP4kqK0J2qxq\nn4RgqsahDYVvTH9w7jXbyLeiNdd8XM2w9U/t7y0Fff/
→ 9yi0GE44Za4rF2LN9d11TPA\nnmRGunUHBcnWEvgJBQ19nJEiU0Zsnvgc/
→ ubhPgXRR4Xq37Z0j4r7g1SgEEzwxA57d\nnemyPxgcYxn/eR44/KJ4EBs+lVDR3veyJm+kXQ99b21/
→ +jh5Xos1AnX5iItreGCc=\n-----END CERTIFICATE-----"
},
"wapm_onion_url": "",
"wapm_url": ""
},
"liquid": {
"address_explorer_url": "https://blockstream.info/liquid/address/",
"asset_registry_onion_url": "http://
→ lhquhzzpzg5tyymcqep24fynpzzqgg3m3rlh7ascnw5cpqsro35bfxyd.onion",
"asset_registry_url": "https://assets.blockstream.info",
"bech32_prefix": "ex",
"bip21_prefix": "liquidnetwork",
"blech32_prefix": "lq",
"blinded_prefix": 12,
"blob_server_onion_url": "",
"blob_server_url": "",
"csv_buckets": [
65535
],
"development": false,
"electrum_onion_url": "explorerzydxu5ecjrkwcayqybizmpjjznk5izmitf2modhcusuqlid.
→ onion:195",

```

(continues on next page)

(continued from previous page)

```

    "electrum_tls": true,
    "electrum_url": "blockstream.info:995",
    "liquid": true,
    "mainnet": true,
    "max_reorg_blocks": 2,
    "name": "Liquid",
    "network": "liquid",
    "p2pkh_version": 57,
    "p2sh_version": 39,
    "pin_server_onion_url": "http://
→ mrrxtq6tjpbnbm7vh5jt6mpjctn7ggyfy5wegvbeff3x7jrznqawlmid.onion",
    "pin_server_public_key":
→ "0332b7b1348bde8ca4b46b9dcc30320e140ca26428160a27bdbfc30b34ec87c547",
    "pin_server_url": "https://jadepin.blockstream.com",
    "policy_asset": "6f0279e9ed041c3d710a9f57d0c02928416460c4b722ae3457a11eec381c526d
→ ",
    "price_onion_url": "http://
→ greenv32e5p4rax6dmfqb4zz17kq2fbmizd7miyava2actplmipyx2qd.onion/prices",
    "price_url": "https://green-bitcoin-mainnet.blockstream.com/prices",
    "server_type": "green",
    "service_chain_code":
→ "02721cc509aa0c2f4a90628e9da0391b196abeabc6393ed4789dd6222c43c489",
    "service_pubkey":
→ "02c408c3bb8a3d526103fb93246f54897bdd997904d3e18295b49a26965cb41b7f",
    "spv_enabled": false,
    "spv_multi": false,
    "spv_servers": [],
    "tx_explorer_url": "https://blockstream.info/liquid/tx/",
    "wamp_cert_pins": [
        "46494e30379059df18be52124305e606fc59070e5b21076ce113954b60517cda",
        "b42688d73bac5099d9cf4fdb7b05f5e54e98c5aa8ab56ee06c297a9a84d2d5f1",
        "67add1166b020ae61b8f5fc96813c04c2aa589960796865572a3c7e737613dfd",
        "64e286b76063602a372efd60cde8db2656a49ee15e84254b3d6eb5fe38f4288b"
    ],
    "wamp_cert_roots": [
        "\n-----BEGIN CERTIFICATE-----\nMIICXjCCAk2gAwIBAgIRAL093/
→ inhFu86QOgQTWzSkUwCgYIKoZIzj0EAwMwTzEL\nMAkGA1UEBhMCVVMxKTAnBgNVBAoTIEludGVybmV0IFNlY3VyaXR5IFJlc2V
→ xwLCA/TmC9N/A8AmzfwdAVXMpcuBe8qQyWj\n+240JxP2T35p0wKZXusKR5LBJJvmsSGPwSSB/
→ GjMH2m6WPUZlvd0xhajggEIMIIB\nBDAOBgNVHQ8BAf8EBAMCAYYwHQYDVR0lBBYwFAYIKwYBBQUHAWIGCCsGAQUFBwMB\n\nMBI
→ wQIMAYBAf8CAQAwHQYDVR0OBBYEFFrz7Sv8NsI3ebLSMOpUb89V\nnyy6sMB8GA1UdIwQYMBaAFHxC1q7eS0g7+pL4nozPbYupc
→ AwCd89eYhN1LxR3pD5xcSAQh8suzYHK019YST8pE9kLJ03uGqQIwWrGxtO3q\nYJkgsTgDyJ2gJrjubi1K9sZnHzOa25JK1fUp
→ Lgul/\n-----END CERTIFICATE-----",
        "\n-----BEGIN CERTIFICATE-----\nMIICXjCCAkygAwIBAgIQTtI99q9+x/
→ mwXHJv+VEqdzAKBggqhkJOPQDazBPMQsw\nCQYDVQQGEwJVUzEpMCCGA1UEChMGSW50ZXJuZXQgU2VjdXJpdHkgUmVzZWZyY2Q
→ LbBVDxCsnsI7vuf4xp4Cu+ETNggGE\nnyBqJ3j8iUwe5Yt/qfSrRf1/
→ D5R58duaJ+IvLRXeASRqEL+VkDXrW3qOCAQgwggEE\nnMA4GA1UdDwEB/
→ wQEAwIBhjAdBgNVHSUEFjAUBgggrBgEFBQcDAgYIKwYBBQUHAWAwEw\nnEgYDVR0TAQH/BAgwBgEB/
→ wIBADAdBgNVHQ4EFgQUbZkq9U0C6+MRwWC6km+NPS7x\nn6kQwHwYDVR0lBBBgwFoAUfEKWrt5LSDv6kviejM9ti6lyN5UwMgYIKw
→ No+KQ\nnUAM1u34E36neL/Zba7ombkIOchSgx1iVxztqFWGddgoG+tpPRPWuhhn\n\n-----END
→ CERTIFICATE-----",
        "\n-----BEGIN CERTIFICATE-----
→ \nMIIFWjCCAOKgAwIBAgIQbkepxUtHDA3sM9CJuRz04TANBgkqhkiG9w0BAQwFADBH\nnMQswCQYDVQQGEwJVUzEiMCAgA1UEChM
→ vo27xqLpeXo4xL+Sv2sfnOhB2x+cWX3u+58qPpvBKJXqeUqv4IyflPLGcY9vX\nnmX7wCl7raKb0xlpHdu0QM+NOsR0jyBhsS+
→ SCYe7\nnzUjwTcLceoiKu7rPWRnWr4+wB7CeMfGCwCdfLqZtbBkOtdh+JhpFAz2weaSUKK0P\nnfylbqAj+lug8aJRT7oM6iCsVl
→ kB+Xj9elx3+naH+uzfsQ551Ve0vSbvlGHR6xYku44LtcXFilWr06zqkUsp\nnzBmkMiVOKvF1RNACzqrOSbTqn3yDsEB750Orp2y
→ VjsPOS+C12LOO\nnRc92wO1AK/
→ 1TD7Cn1TsNsYqiA94xrc36m97PtbfkSIS5r762DL8EGMUUXLeXdyW\nnk70paDPvOmsB4om3pXV2V4J95eSRQAogB/
→ mqghtcmxlbCluQ0WEdrHbEg8QOB+\nDVRNVjzRlW5y0vtOUucxD/SVRNuJLDWcfr0wbrM
→ oFB2ACYPTrIrnqYXngF\nnLQIDAQABoIwQDAOBgNVHQ8BAf8EBAMCAQYwDwYDVR0TAQH/BAUwAwEB/
→ zAdBgNV\nnHQ4EFgQU5K8rJnEak0gnhs9S5zizv8IkTcT4wDQYJKoZIhvcNAQEMBQADggIBADiW\nnCu49tLJYeX++dnAsznyvgyv3
74UEBZvXw6Z\nnXPYfcX3v73svfu021pdwCxxu11xWaj0140k4DLh9+42FpLFZXvRq4d2n
6xmld01D1zvICxi/ZG6qcZ8WpyTgYmpl0p8WnK00dC3\nnd8t5/
→ Wk6k jftbjhlRn7pYL15iJdfOBL07q9bgsiG1eGZbYwE8na6SfZu6W0eX6Dv\nnJ4J2QPim01hcDyxC2kLGe4g0x8HYRZvBPsVhH
→ Zg\nnDdIX020YI5NAaIFItO/
→ Nj355-236-Na-237-238-239-240-241-242-243-244-245-246-247-248-249-250-251-252-253-254-255-256-257-258-259-260-261-262-263-264-265-266-267-268-269-270-271-272-273-274-275-276-277-278-279-280-281-282-283-284-285-286-287-288-289-290-291-292-293-294-295-296-297-298-299-300-301-302-303-304-305-306-307-308-309-310-311-312-313-314-315-316-317-318-319-320-321-322-323-324-325-326-327-328-329-330-331-332-333-334-335-336-337-338-339-340-341-342-343-344-345-346-347-348-349-350-351-352-353-354-355-356-357-358-359-360-361-362-363-364-365-366-367-368-369-370-371-372-373-374-375-376-377-378-379-380-381-382-383-384-385-386-387-388-389-390-391-392-393-394-395-396-397-398-399-400-401-402-403-404-405-406-407-408-409-410-411-412-413-414-415-416-417-418-419-420-421-422-423-424-425-426-427-428-429-430-431-432-433-434-435-436-437-438-439-440-441-442-443-444-445-446-447-448-449-450-451-452-453-454-455-456-457-458-459-460-461-462-463-464-465-466-467-468-469-470-471-472-473-474-475-476-477-478-479-480-481-482-483-484-485-486-487-488-489-490-491-492-493-494-495-496-497-498-499-500-501-502-503-504-505-506-507-508-509-510-511-512-513-514-515-516-517-518-519-520-521-522-523-524-525-526-527-528-529-530-531-532-533-534-535-536-537-538-539-540-541-542-543-544-545-546-547-548-549-550-551-552-553-554-555-556-557-558-559-560-561-562-563-564-565-566-567-568-569-570-571-572-573-574-575-576-577-578-579-580-581-582-583-584-585-586-587-588-589-590-591-592-593-594-595-596-597-598-599-600-601-602-603-604-605-606-607-608-609-610-611-612-613-614-615-616-617-618-619-620-621-622-623-624-625-626-627-628-629-630-631-632-633-634-635-636-637-638-639-640-641-642-643-644-645-646-647-648-649-650-651-652-653-654-655-656-657-658-659-660-661-662-663-664-665-666-667-668-669-670-671-672-673-674-675-676-677-678-679-680-681-682-683-684-685-686-687-688-689-690-691-692-693-694-695-696-697-698-699-700-701-702-703-704-705-706-707-708-709-710-711-712-713-714-715-716-717-718-719-720-721-722-723-724-725-726-727-728-729-730-731-732-733-734-735-736-737-738-739-740-741-742-743-744-745-746-747-748-749-750-751-752-753-754-755-756-757-758-759-760-761-762-763-764-765-766-767-768-769-770-771-772-773-774-775-776-777-778-779-780-781-782-783-784-785-786-787-788-789-790-791-792-793-794-795-796-797-798-799-800-801-802-803-804-805-806-807-808-809-810-811-812-813-814-815-816-817-818-819-820-821-822-823-824-825-826-827-828-829-830-831-832-833-834-835-836-837-838-839-840-841-842-843-844-845-846-847-848-849-850-851-852-853-854-855-856-857-858-859-860-861-862-863-864-865-866-867-868-869-870-871-872-873-874-875-876-877-878-879-880-881-882-883-884-885-886-887-888-889-890-891-892-893-894-895-896-897-898-899-900-901-902-903-904-905-906-907-908-909-910-911-912-913-914-915-916-917-918-919-920-921-922-923-924-925-926-927-928-929-930-931-932-933-934-935-936-937-938-939-940-941-942-943-944-945-946-947-948-949-950-951-952-953-954-955-956-957-958-959-960-961-962-963-964-965-966-967-968-969-970-971-972-973-974-975-976-977-978-979-980-981-982-983-984-985-986-987-988-989-990-991-992-993-994-995-996-997-998-999-1000-1001-1002-1003-1004-1005-1006-1007-1008-1009-1010-1011-1012-1013-1014-1015-1016-1017-1018-1019-1020-1021-1022-1023-1024-1025-1026-1027-1028-1029-1030-1031-1032-1033-1034-1035-1036-1037-1038-1039-1040-1041-1042-1043-1044-1045-1046-1047-1048-1049-1050-1051-1052-1053-1054-1055-1056-1057-1058-1059-1060-1061-1062-1063-1064-1065-1066-1067-1068-1069-1070-1071-1072-1073-1074-1075-1076-1077-1078-1079-1080-1081-1082-1083-1084-1085-1086-1087-1088-1089-1090-1091-1092-1093-1094-1095-1096-1097-1098-1099-1100-1101-1102-1103-1104-1105-1106-1107-1108-1109-1110-1111-1112-1113-1114-1115-1116-1117-1118-1119-1120-1121-1122-1123-1124-1125-1126-1127-1128-1129-1130-1131-1132-1133-1134-1135-1136-1137-1138-1139-1140-1141-1142-1143-1144-1145-1146-1147-1148-1149-1150-1151-1152-1153-1154-1155-1156-1157-1158-1159-1160-1161-1162-1163-1164-1165-1166-1167-1168-1169-1170-1171-1172-1173-1174-1175-1176-1177-1178-1179-1180-1181-1182-1183-1184-1185-1186-1187-1188-1189-1190-1191-1192-1193-1194-1195-1196-1197-1198-1199-1200-1201-1202-1203-1204-1205-1206-1207-1208-1209-1210-1211-1212-1213-1214-1215-1216-1217-1218-1219-1220-1221-1222-1223-1224-1225-1226-1227-1228-1229-1230-1231-1232-1233-1234-1235-1236-1237-1238-1239-1240-1241-1242-1243-1244-1245-1246-1247-1248-1249-1250-1251-1252-1253-1254-1255-1256-1257-1258-1259-1260-1261-1262-1263-1264-1265-1266-1267-1268-1269-1270-1271-1272-1273-1274-1275-1276-1277-1278-1279-1280-1281-1282-1283-1284-1285-1286-1287-1288-1289-1290-1291-1292-1293-1294-1295-1296-1297-1298-1299-1300-1301-1302-1303-1304-1305-1306-1307-1308-1309-1310-1311-1312-1313-1314-1315-1316-1317-1318-1319-1320-1321-1322-1323-1324-1325-1326-1327-1328-1329-1330-1331-1332-1333-1334-1335-1336-1337-1338-1339-1340-1341-1342-1343-1344-1345-1346-1347-1348-1349-1350-1351-1352-1353-1354-1355-1356-1357-1358-1359-1360-1361-1362-1363-1364-1365-1366-1367-1368-1369-1370-1371-1372-1373-1374-1375-1376-1377-1378-1379-1380-1381-1382-1383-1384-1385-1386-1387-1388-1389-1390-1391-1392-1393-1394-1395-1396-1397-1398-1399-1400-1401-1402-1403-1404-1405-1406-1407-1408-1409-1410-1411-1412-1413-1414-1415-1416-1417-1418-1419-1420-1421-1422-1423-1424-1425-1426-1427-1428-1429-1430-1431-1432-1433-1434-1435-1436-1437-1438-1439-1440-1441-1442-1443-1444-1445-1446-1447-1448-1449-1450-1451-1452-1453-1454-1455-1456-1457-1458-1459-1460-1461-1462-1463-1464-1465-1466-1467-1468-1469-1470-1471-1472-1473-1474-1475-1476-1477-1478-1479-1480-1481-1482-1483-1484-1485-1486-1487-1488-1489-1490-1491-1492-1493-1494-1495-1496-1497-1498-1499-1500-1501-1502-1503-1504-1505-1506-1507-1508-1509-1510-1511-1512-1513-1514-1515-1516-1517-1518-1519-1520-1521-1522-1523-1524-1525-1526-1527-1528-1529-1530-1531-1532-1533-1534-1535-1536-1537-1538-1539-1540-1541-1542-1543-1544-1545-1546-1547-1548-1549-1550-1551-1552-1553-1554-1555-1556-1557-1558-1559-1560-1561-1562-1563-1564-1565-1566-1567-1568-1569-1570-1571-1572-1573-1574-1575-1576-1577-1578-1579-1580-1581-1582-1583-1584-1585-1586-1587-1588-1589-1590-1591-1592-1593-1594-1595-1596-1597-1598-1599-1600-1601-1602-1603-1604-1605-1606-1607-1608-1609-1610-1611-1612-1613-1614-1615-1616-1617-1618-1619-1620-1621-1622-1623-1624-1625-1626-1627-1628-1629-1630-1631-1632-1633-1634-1635-1636-1637-1638-1639-1640-1641-1642-1643-1644-1645-1646-1647-1648-1649-1650-1651-1652-1653-1654-1655-1656-1657-1658-1659-1660-1661-1662-1663-1664-1665-1666-1667-1668-1669-1670-1671-1672-1673-1674-1675-1676-1677-1678-1679-1680-1681-1682-1683-1684-1685-1686-1687-1688-1689-1690-1691-1692-1693-1694-1695-1696-1697-1698-1699-1700-1701-1702-1703-1704-1705-1706-1707-1708-1709-1710-1711-1712-1713-1714-1715-1716-1717-1718-1719-1720-1721-1722-1723-1724-1725-1726-1727-1728-1729-1730-1731-1732-1733-1734-1735-1736-1737-1738-1739-1740-1741-1742-1743-1744-1745-1746-1747-1748-1749-1750-1751-1752-1753-1754-1755-1756-1757-1758-1759-1760-1761-1762-1763-1764-1765-1766-1767-1768-1769-1770-1771-1772-1773-1774-1775-1776-1777-1778-1779-1780-1781-1782-1783-1784-1785-1786-1787-1788-1789-1790-1791-1792-1793-1794-1795-1796-1797-1798-1799-1800-1801-1802-1803-1804-1805-1806-1807-1808-1809-1810-1811-1812-1813-1814-1815-1816-1817-1818-1819-1820-1821-1822-1823-1824-1825-1826-1827-1828-1829-1830-1831-1832-1833-1834-1835-1836-1837-1838-1839-1840-1841-1842-1843-1844-1845-1846-1847-1848-1849-1850-1851-1852-1853-1854-1855-1856-1857-1858-1859-1860-1861-1862-1863-1864-1865-1866-1867-1868-1869-1870-1871-1872-1873-1874-1875-1876-1877-1878-1879-1880-1881-1882-1883-1884-1885-1886-1887-1888-1889-1890-1891-1892-1893-1894-1895-1896-1897-1898-1899-1900-1901-1902-1903-1904-1905-1906-1907-1908-1909-1910-1911-1912-1913-1914-1915-1916-1917-1918-1919-1920-1921-1922-1923-1924-1925-1926-1927-1928-1929-1930-1931-1932-1933-1934-1935-1936-1937-1938-1939-1940-1941-1942-1943-1944-1945-1946-1947-1948-1949-1950-1951-1952-1953-1954-1955-1956-1957-1958-1959-1960-1961-1962-1963-1964-1965-1966-1967-1968-1969-1970-1971-1972-1973-1974-1975-1976-1977-1978-1979-1980-1981-1982-1983-1984-1985-1986-1987-1988-1989-1990-1991-1992-1993-1994-1995-1996-1997-1998-1999-2000-2001-2002-2003-2004-2005-2006-2007-2008-2009-2010-2011-2012-2013-2014-2015-2016-2017-2018-2019-2020-2021-2022-2023-2024-2025-2026-2027-2028-2029-2030-2031-2032-2033-2034-2035-2036-2037-2038-2039-2040-2041-2042-2043-2044-2045-2046-2047-2048-2049-2050-2051-2052-2053-2054-2055-2056-2057-2058-2059-2060-2061-2062-2063-2064-2065-2066-2067-2068-2069-2070-2071-2072-2073-2074-2075-2076-2077-2078-2079-2080-2081-2082-2083-2084-2085-2086-2087-2088-2089-2090-2091-2092-2093-2094-2095-2096-2097-2098-2099-2100-2101-2102-2103-2104-2105-2106-2107-2108-2109-2110-2111-2112-2113-2114-2115-2116-2117-2118-2119-2120-2121-2122-2123-2124-2125-2126-2127-2128-2129-2130-2131-2132-2133-2134-2135-2136-2137-2138-2139-2140-2141-2142-2143-2144-2145-2146-2147-2148-2149-2150-2151-2152-2153-2154-2155-2156-2157-2158-2159-2160-2161-2162-2163-2164-2165-2166-2167-2168-2169-2170-2171-2172-2173-2174-2175-2176-2177-2178-2179-2180-2181-2182-2183-2184-2185-2186-2187-2188-2189-2190-2191-2192-2193-2194-2195-2196-2197-2198-2199-2200-2201-2202-2203-2204-2205-2206-2207-2208-2209-2210-2211-2212-2213-2214-2215-2216-2217-2218-2219-2220-2221-2222-2223-2224-2225-2226-2227-2228-2229-2230-2231-2232-2233-2234-2235-2236-2237-2238-2239-2240-2241-2242-2243-2244-2245-2246-2247-2248-2249-2250-2251-2252-2253-2254-2255-2256-2257-2258-2259-2260-2261-2262-2263-2264-2265-2266-2267-2268-2269-2270-2271-2272-2273-2274-2275-2276-2277-2278-2279-2280-2281-2282-2283-2284-2285-2286-2287-2288-2289-2290-2291-2292-2293-2294-2295-2296-2297-2298-2299-2300-2301-2302-2303-2304-2305-2306-2307-2308-2309-2310-2311-2312-2313-2314-2315-2316-2317-2318-2319-2320-2321-2322-2323-2324-2325-2326-2327-2328-2329-2330-2331-2332-2333-2334-2335-2336-2337-2338-2339-2340-2341-2342-2343-2344-2345-2346-2347-2348-2349-2350-2351-2352-2353-2354-2355-2356-2357-2358-2359-2360-2361-2362-2363-2364-2365-2366-2367-
```

(continued from previous page)

```

"\n-----BEGIN CERTIFICATE-----
↪ \nMIIFazCCA1OgAwIBAgIRAIQz7DSQONZRGPGu2OCiwAwDQYJKoZIhvcNAQELBQAw\nTzELMAkGA1UEBhMCVVMxKTAnBgNVBA
↪ vVqbnvYATyjb3miGbESTtrFj/RQSa78f0uoxmyF+\n0TM8ukj13Xnfs7j/
↪ EvEhmkvBioZxaUpmZmyPfjxwv60pIgbz5MDmgK7iS4+3mX6U\nnA5/
↪ TR5d8mUgjU+g4rk8Kb4Mu0U1XjIB0ttov0DiNewNwIRt18jA8+o+u3dpjq+sW\nnT8KOEUt+zwvo/
↪ 7V3LvSye0rgTBiLDHCNAymg4VMk7BPZ7hm/
↪ ELNKjD+Jo2FR3qyH\nb5T0Y3HsLuJvW5iB4Y1cNHlsdu87kGJ55tukmi8mxdAQ4Q7e2RCOFvu396j3x+UC\nb5iPngiV5+I31g
↪ lJBdiB3QW0ktZB6awBdpUKD9jflb0SHzUv\nKBds0pjBqAlkd25HN7rOrFleaJ1/
↪ ctaJxQZBKT5ZPt0m9STJEadao0xAH0ahmbWn\nOlFuhjuefXKnEgV4We0+UXgVCwOPjdAvBbI+e0ocS3MFEVzG6uBQE3xDk3Szy
↪ rOPNk3sgrDQoo//fb4hVC1CLQJ13hef4Y53CI\nnrU7m2Ys6xt0nUW7/
↪ vGT1M0NPAGMBAAGjQjBAMA4GA1UdDwEB/wQEAWIBBjAPBgNV\nnHRMBAf8EBTADAQH/
↪ MB0GA1UdDgQWBRR5tFnme7b15AFzgAiIyBpY9umbbjANBgkq\nnhkiG9w0BAQsFAAOCAGEAVR9YqbyyqFDQDLHYGmkGJykIrGF1
↪ V91ZL\nubhzEFnTIZd+50xx+7LSYK05qAvqFyFWhfQDlnrzuBZ6brJFe+GnY+EgPbk6ZGQ\nn3BebYhtF8GaV0nxvwuo77x/
↪ Py9auJ/GpsMiu/X1+mvoiBOv/2X/qkSsisRcOj/
↪ KK\nnNftY2PwByVS5uCbmioGziUwthDyC3+6WVwW6LLv3xLfHTjuCvjHIInNzktHCgKQ5\nnORAZI4JMPJ+GslWYHb4phowim57ia
↪ +sKAiuvtd7u+Nxe5AW0wdeRlN8NwdC\nnjNPElpzVmbUq4JUagEiuTDkHszxHpFKVK7q4+63SM1N95R1NbdWhscdCb+ZAJzVc\nn
↪ l1vh+wjChP4kqK0J2qxq\nn4RgqsahDYVvTH9w7jXbyLeiNdd8XM2w9U/t7y0Ff/
↪ 9yi0GE44Za4rF2LN9d11TPA\nnmRGunUHBcnWEvgJBQ19nJEiU0Zsnvgc/
↪ ubhPgXRR4Xq37Z0j4r7g1SgEEzwxA57d\nnemyPxgcYxn/eR44/KJ4EBs+lVDR3veyJm+kXQ99b21/
↪ +jh5Xos1AnX5iItreGCc=\n-----END CERTIFICATE-----"
    ],
    "wamp_onion_url": "ws://liquidbtccgecscpokecnr5uwg2de55shdq7dnvlpzeju7tnefbekicqd.
↪ onion/v2/ws",
    "wamp_url": "wss://green-liquid-mainnet.blockstream.com/v2/ws"
  },
  "localtest": {
    "address_explorer_url": "",
    "bech32_prefix": "bcr",
    "bip21_prefix": "bitcoin",
    "blob_server_onion_url": "",
    "blob_server_url": "",
    "csv_buckets": [
      144,
      4320,
      51840
    ],
    "development": true,
    "electrum_onion_url": "",
    "electrum_tls": false,
    "electrum_url": "localhost:19002",
    "liquid": false,
    "mainnet": false,
    "max_reorg_blocks": 1008,
    "name": "Localtest",
    "network": "localtest",
    "p2pkh_version": 111,
    "p2sh_version": 196,
    "pin_server_onion_url": "http://
↪ mrrxtq6tjpbnbm7vh5jt6mpjctn7ggyfy5wegvbeff3x7jrznqawlmid.onion",
    "pin_server_public_key":
↪ "0332b7b1348bde8ca4b46b9dcc30320e140ca26428160a27bdbfc30b34ec87c547",
    "pin_server_url": "https://jadepin.blockstream.com",
    "price_onion_url": "",
    "price_url": "http://localhost:8081",
    "server_type": "green",
    "service_chain_code":
↪ "b60befcc619bb1c212732770fe181f2f1aa824ab89f8aab49f2e13e3a56f0f04",

```

(continues on next page)

(continued from previous page)

```

"service_pubkey":
→ "036307e560072ed6ce0aa5465534fb5c258a2ccfbcb257f369e8e7a181b16d897b3",
"spv_enabled": false,
"spv_multi": false,
"spv_servers": [],
"tx_explorer_url": "",
"wamp_cert_pins": [],
"wamp_cert_roots": [
  "\n-----BEGIN CERTIFICATE-----\nMIICXjCCAk2GwIBAgIRALO93/
→ inhFu86QOgQTWzSkUwCgYIKoZIj0EAwMwTzEL\nMAkGAlUEBhMCMVVMxKTAnBgNVBAoTIEludGVybmV0IFNlY3VyaXR5IFJlc2V
→ xwLcA/TmC9N/A8AmzfdAVXMpcuBe8qQyWj\n+240JxP2T35p0wKZXuskR5LBJJvmsSGPwSSB/
→ GjMH2m6WPUZiVd0xhajggEIMIIB\nBDAOBgNVHQ8BAf8EBAMCAYYwHQYDVR01BBYwFAYIKwYBBQUHAWIGCCsGAQUFBwMB\n→ wQIMAYBAf8CAQAwHQYDVR0OBBYEFFrz7Sv8NsI3eb1SM0pUb89V\nnyy6sMB8GA1UdIwQYMBaAFHxC1q7eS0g7+pL4nozPbYupc
→ AwCd89eYhN1XLr3pD5xcSAQh8suzYHK0l9YST8pE9kLJ03uGqQIwWrGxtO3q\nyYJkgsTgDyJ2gJrjubi1K9sZmHzOa25JK1fUpl
→ Lgul/\n-----END CERTIFICATE-----",
  "\n-----BEGIN CERTIFICATE-----\nMIICXjCCAkYgAwIBAgIQTtI99q9+x/
→ mwXhJjv+VEqdzAKBggqhkjOPQQDAzBPMQsw\nnCQYDVQQGEwJVUzEpMCCGA1UEChMgSW50ZXJ1ZXQgU2VjdXJpdHkgUmVzZWZyY2
→ LbBVDxCSnsI7vuf4xp4Cu+ETNggGE\nnyBqJ3j8iUwe5Yt/qfSrRf1/
→ D5R58duaJ+IvLRXeASRqEL+VkDXrW3qOCAQgwggEE\nnMA4GA1UdDwEB/
→ wQEAWIBhjAdBgNVHSUEFjAUBgggrBgEFBQcDAgYIKwYBBQUHAWew\nnEgYDVROTAQH/BAgwBgEB/
→ wIBADAdBgNVHQ4EFgQUbZkq9U0C6+MRwWC6km+NPS7x\nn6kQwHwYDVRO1jBBgwFoAUfEKWrt5LSDv6kviejm9ti6lyN5UwMgYIKv
→ No+KQ\nnUAM1u34E36neL/Zba7ombkIOchSgxliVxztqFWGddgoG+tpPRPWhuhhn\n-----END
→ CERTIFICATE-----",
  "\n-----BEGIN CERTIFICATE-----
→ \nMIIFWjCCAOKgAwIBAgIQbkepxUtHDA3sM9CJuRz04TANBgkqhkiG9w0BAQwFADBH\nnMQswCQYDVQQGEwJVUzEiMCAgAlUEChM
→ vo27xqLpeXo4xL+Sv2sfnohB2x+cWX3u+58qPpvBKJXqeqUqv4IyflpLgCY9vX\nnmX7wCl7raKb0xlpHdu0QM+NoSROjyBhsS+
→ SCYe7\nnzUjwTcLceoiKu7rPWRnWr4+wB7CeMfGCwcdLfQZtbBkOtdh+JhpFaz2weaSUKKOP\nnfylblqAj+lug8aJRT7oM6iCsVlq
→ kB+Xj9elx3+naH+uzfsQ55lVe0vSbvlghR6xYku44LtcXFilWr06zqkUsp\nnzBmkMiVOKvFlRNACzqrOSbtqn3yDsEB7500r2y
→ VjsPOS+C12LOO\nnRc92w01AK/
→ 1TD7Cn1TsNsYqiA94xrcx36m97PtbfkSIS5r762DL8EGMUUXLeXdYW\nnk70paDPvOmbS4om3xPXV2V4J95eSRQAogB/
→ mgqhtqmxlbCluQ0WEdrHbEg8QOB+\nndVrNVjzRlw5y0vtOUucXD/SVRNuJLDWcfr0wbrM7Rv1/
→ oFB2ACYPTIrInqYXngF\nnlQIDAQABO0IwQDAOBgNVHQ8BAf8EBAMCAQYwDwYDVROTAQH/BAUwAwEB/
→ zAdBgNV\nnHQ4EFgQU5K8rJnEaK0gnhS9SZizv8IkTcT4wDQYJKoZIhvcNAQEMBQADggIBADiW\nnCu49tJYeX++dnAsznyvgvy3S
→ UEBZvXw6Z\nnXPYfcX3v73svfu021pdwCxxu11xWaJ0140k4DLh9+42FpLFZxvRq4d2h9mReruZR\nngyFmxhE+885H7pwoHyXa/
→ 6xmld01D1zviCxi/ZG6qcz8WpyTgYmPl0p8WnK00dC3\nnd8t5/
→ Wk6kjftbjhlRn7pYL15iJdfOBL07q9bgsiG1eGZbYwE8na6SfZu6W0eX6Dv\nnJ4J2QPim01hcDyx2kLGe4g0x8HYRZvBPsVhH
→ Zg\nnDdIX02OYI5NaAIFItO/
→ Nis3Jz5nu2Z6qNuFoS3FJFDYoOj0dzpqPJeaAcWErtXvM\nn+SUWgeExX6GjfhaknBZqlxi9dnKlC54dNuYvoS++cJEPqOba+MS
→ ZRBils6Kl0obsXs7X9\nnSQ98POyDGCBDTtWTurQ0sR8WNh8M5mQ5Fkzc4P4dyKliPUDqysU0ArSuiYgzNdws\nnE3PYJ/
→ HQcu51OyLemGhmW/HGY0dVHLq1CFF1pkl\nn-----END CERTIFICATE-----",
  "\n-----BEGIN CERTIFICATE-----
→ \nMIIFazCCA1OgAwIBAgIRAIQz7DSQONZRGpGu2OCiAwDQYJKoZIhvcNAQELBQA\nnTzELMAkGAlUEBhMCMVVMxKTAnBgNVBAoTIEludGVybmV0IFNlY3VyaXR5IFJlc2V
→ vVqbvYATyjb3miGbESTtrFj/RQSa78f0uoxmyF+\nnoTM8ukj13Xnfs7j/
→ EvEhmkvBioZxaUpmZmyPfjxwv60pIgbz5MDmgK7iS4+3mX6U\nnA5/
→ TR5d8mUgjU+g4rk8Kb4Mu0U1XjIB0ttov0DiNewNwIRt18jA8+o+u3dpjq+sW\nnT8KOEut+zwvo/
→ 7V3LvSye0rgTBI1DHCNAymg4VMK7BPZ7hm/
→ ELNKjD+Jo2FR3qyH\nnB5TOY3HsLuJvW5iB4Y1cNHlsdu87kGJ55tukmi8mxdAQ4Q7e2RCOFvu396j3x+UC\nnB5iPNgiV5+I31g
→ 1JBdiB3QW0ktZB6awBdpUKD9jflb0SHzUv\nnKBds0pjBqAlkd25HN7rOrFleaJ1/
→ ctaJxQZBKT5ZPt0m9STJEadao0xAH0ahmbWn\nnOlFuhjuefXKNEgV4We0+UXgVCwOPjdAvBbI+e0ocS3MFEVzG6uBQE3xDk3Sz
→ rOPnk3sgrDQoo//fb4hVC1CLQJ13hef4Y53CI\nnrU7m2Ys6xt0nUW7/
→ vGT1M0NPagMBAAGjQjBAMA4GA1UdDwEB/wQEAWIBBjAPBgNV\nnHRMBAf8EBTADAQH/
→ MB0GA1UdDgQWBBR5tFnme7b15AFzgAiIyBpY9umbbjANBgkq\nnhkiG9w0BAQsFAAOCAgEAVR9YqbyyqFDQDLHYGmkGjYkIrGF1
→ V91ZL\nnubhzEFnTIZd+50xx+7LSYK05gAvqFyFWhfQDlnrzuBZ6brJFe+GnY+EGPbk6ZGQ\nn3BebYhtF8GaV0nxvwuo77x/
→ Py9auJ/GpsMiu/X1+mvoiBOv/2X/qkSsisRcOj/
→ KK\nnNftY2PwByVS5UcBmzogziUwthDyC3+6WVwW6LLv3xLfHTjuCvJHIInNzktHCgKQ5\nnORAZI4JMPJ+GslWYHb4phowim57ia
→ +sKAIuvtd7u+Nxe5AW0wderlN8NwdC\nnjNPElpzVmbUq4JUagEiuTDkHszxHpFKVK7q4+63SM1N95R1NbdWhscdCb+ZAJzVc\nno
→ 1lvh+wjChP4kqK0J2qxq\nn4RgqsahDYVvTH9w7jXbyLeiNddXM2w9U/t7y0Ff/
→ 9yi0GE44Za4rF2LN9d11TPA\nnmRGunUHBcnWEvgJBQ19nJEiU0Zsnvgc/
→ ubhPgXRR4Xq37Z0j4r7g1SgEEzwxA57d\nnemyPxcgYxn/eR44/KJ4EBs+lVDR3veyJm+kXQ99b21/
→ +jh5Xos1AnX5iitreGCc=\n-----END CERTIFICATE-----"

```

(continues on next page)

(continued from previous page)

```

    },
    "wamp_onion_url": "",
    "wamp_url": "ws://localhost:8080/v2/ws"
  },
  "localtest-liquid": {
    "address_explorer_url": "",
    "asset_registry_onion_url": "http://
→lhquhzzpzg5tyymcqep24fynpzzqgg3m3rlh7ascnw5cpqsro35bfxyd.onion",
    "asset_registry_url": "https://assets.blockstream.info",
    "bech32_prefix": "ert",
    "bip21_prefix": "liquidnetwork",
    "blech32_prefix": "el",
    "blinded_prefix": 4,
    "blob_server_onion_url": "",
    "blob_server_url": "",
    "csv_buckets": [
      1440,
      65535
    ],
    "development": true,
    "electrum_onion_url": "",
    "electrum_tls": false,
    "electrum_url": "localhost:19002",
    "liquid": true,
    "mainnet": false,
    "max_reorg_blocks": 2,
    "name": "Localtest Liquid",
    "network": "localtest-liquid",
    "p2pkh_version": 235,
    "p2sh_version": 75,
    "pin_server_onion_url": "http://
→mrrxtq6tjpbnbm7vh5jt6mpjctn7ggyfy5wegvbef3x7jrznqawlmid.onion",
    "pin_server_public_key":
→"0332b7b1348bde8ca4b46b9dcc30320e140ca26428160a27bdbfc30b34ec87c547",
    "pin_server_url": "https://jadenpin.blockstream.com",
    "policy_asset": "5ac9f65c0efcc4775e0baec4ec03abdde22473cd3cf33c0419ca290e0751b225
→",
    "price_onion_url": "",
    "price_url": "http://localhost:8081",
    "server_type": "green",
    "service_chain_code":
→"b60befcc619bb1c212732770fe181f2f1aa824ab89f8aab49f2e13e3a56f0f04",
    "service_pubkey":
→"036307e560072ed6ce0aa5465534fb5c258a2ccfbc257f369e8e7a181b16d897b3",
    "spv_enabled": false,
    "spv_multi": false,
    "spv_servers": [],
    "tx_explorer_url": "",
    "wamp_cert_pins": [],
    "wamp_cert_roots": [
      "\n-----BEGIN CERTIFICATE-----\nMIICxjCCAk2gAwIBAgIRALO93/
→inhFu86QOgQTWzSkUwCgYIKoZIzj0EAwMwTzEL\nMAkGA1UEBhMCVVMxKTAnBgNVBAoTIEludGVybmV0IFNlY3VyaXR5IFJlc2
→xwLCA/TmC9N/A8AmzfwdAVXMpcuBe8qQyWj\n+240JxP2T35p0wKZXuskR5LBJJvmsSGPwSSB/
→GjMH2m6WPUZIVd0xhajggEIMIIB\nBDAOBgNVHQ8BAf8EBAMCAYYhQYDVR01BBYwFAYIKwYBBQUHAWIGCCsGAQUFBwMB\n\nMBI
→wQIMAYBAf8CAQAwHqYDVR00BBYEFFrz7Sv8NsI3ebLSMOpUb89V\nnyy6sMB8GA1UdIwQYMBaAFHxC1q7eS0g7+pL4nozPbYupc
→AwCd89eYhN1XLr3pD5xcSAQh8suzYHK019YST8pE9kLJ03uGqQIwWrGxtO3q\nYnYJkgsTgDyJ2gJrjubilK9sZmHzOa25JK1fUp
→Lgul/\n-----END CERTIFICATE-----",

```

(continues on next page)

(continued from previous page)

```

"\n-----BEGIN CERTIFICATE-----\nMIICxjCCAkYgAwIBAgIQTtI99q9+x/
→mwXHJv+VEqdzAKBggqhkJOPQDZBPMQsw\nCQYDVQQGEwJVUzEpMCcGA1UEChMgSW50ZXJuZXQgU2VjdXJpdHkgUmVzZWZyY2
→LbBVDxCsnsI7vuf4xp4Cu+ETNggGE\nyBqJ3j8iUwe5Yt/qfSrRf1/
→D5R58duaJ+IvLRXeASRqEL+VkdXrW3qOCAQgwggEE\nnMA4GA1UdDwEB/
→wQEAwIBhjAdBgNVHSUEFjAUBgggrBgEFBQcDAGYIKWYBBQUHAAwEw\nnEgYDVDR0TAQH/BAGwBgEB/
→wIBADAdBgNVHQ4EFgQUbZkq9U0C6+MRwWC6km+NPS7x\nn6kQwHwYDVDR0jBBgwFoAUfEKWrt5LSDv6kviejm9t161yN5UwMgYIK
→No+KQ\nnUAM1u34E36neL/Zba7ombkIOChSgx1iVxzqtFWGddgoG+tpRPWhuhhn\nn-----END
→CERTIFICATE-----",

"\n-----BEGIN CERTIFICATE-----
→\nMIIFWjCCAOKgAwIBAgIQbkepxUtHDA3sM9CJuRz04TANBgkqhkiG9w0BAQwFADBH\nnMQswCQYDVQQGEwJVUzEiMCAGA1UEChM
→vo27xqLpeXo4xL+Sv2sfnOhB2x+cWX3u+58qPpvBKJXqeqUqv4IyflPLGcY9vX\nnmX7wC17raKb0xlpHdU0QM+NOSROjyBhsS+
→SCYe7\nnzUjwTcLCEoiKu7rPWRnWr4+wB7CeMfGCwcDfLqZtbBkOtdh+JhpFAz2weaSUKKOP\nnfyblqAj+lug8aJRT7oM6iCsV1
→kB+Xj9e1x3+naH+uzfsQ551Ve0vSbvlGHR6xYKu44LtcXFilWr06zqkUsp\nnzBmkMiVOKvFLRNACzqrOSbTqn3yDsEB750Orp2
→VjsPOS+C12LOO\nnRc92wO1AK/
→1TD7Cn1TsNsYqiA94xrcx36m97PtbfkSIS5r762DL8EGMUUXLeXdyW\nnk70paDPvOmsB4om3xPXV2V4J95eSRQAogB/
→mqgghtqmxlbCluQ0WEdrHbEg8QOB+\nndVrNVjzRlw5y0vtOUucD/SVRNuJLDWcfr0wbrM7Rv1/
→oFB2ACYPTTrIrnqYNxgF\nnlQIDAQABO0IwQDAOBGNVHQ8BAf8EBAMCAQYwDwYDVDR0TAQH/BAUwAwEB/
→zAdBgNV\nnHQ4EFgQU5K8rJnEaK0gnhS9S2izv8IkTcT4wDQYJKoZIhvcNAQEMBAQgIBADiW\nnCu49tJYeX++dnAsznyvgv3
→UEBZvXw6Z\nnXPYfcX3v73svfu02lpdwCxu11xWajol40k4DLh9+42FpLFZxvRq4d2h9mREruZR\nngyFmxhE+885H7pwoHyXa/
→6xmld01D1zvICxi/ZG6qcZ8WpyTgYmpl0p8WnK0OdC3\nnd8t5/
→Wk6kjftbjhlRn7pYL15iJdfOBL07q9bgsiG1eGZbYwE8na6SfZu6W0eX6Dv\nnJ4J2QPim01hcDyx2kLGe4g0x8HYRZvBPsVhH
→Zg\nnDdIx02OYI5NaAIFiTo/
→Nis3JZ5nu2Z6qNuFoS3FJFYDo0j0dzpqPJeaAcWErtXvM\nn+SUWgeExX6GjfhaknBZqlxi9dnK1C54dNuYvoS++cJEPqOba+MS
→ZRBiIs6Kl0obsXs7X9\nnSQ98POyDGCBDTtWTurQ0sR8WNh8M5mQ5Fkzc4P4dyKliPUDqysU0ArSuiYgzNdws\nnE3PYJ/
→HQcu51OyLemGhmW/HGY0dVHLq1CFF1pkgl\nn-----END CERTIFICATE-----",

"\n-----BEGIN CERTIFICATE-----
→\nMIIFazCCA1OgAwIBAgIRAIQz7DSQONZRGpGu2OCiwAwDQYJKoZIhvcNAQELBQAw\nnTzELMAkGA1UEBhMCVVMxKTAnBgNVBAQ
→vVqbvYATyjb3miGbESTtrFj/RQSa78f0uoxmyF+\nnoTM8ukj13Xnfs7j/
→EvEhmkvBioZxaUpmZmyPfjxwv60pIgbz5MDmgK7iS4+3mX6U\nnA5/
→TR5d8mUgJ+g4rk8Kb4Mu0U1XjIB0ttov0DiNewNwIRt18jA8+o+u3dpjq+sW\nnT8KOEut+zwvo/
→7V3LvSye0rgTB1LDHCNAymg4VMk7BPZ7hm/
→ELNKjD+Jo2FR3qyH\nnB5T0Y3HsLuJvW5iB4Y1cNHlsdu87kGJ55tukmi8mxdAQ4Q7e2RCOFvu396j3x+UC\nnB5iPNgiV5+I3lg
→1JBdiB3QW0KtZB6awBdpUKD9jflb0SHzUv\nnKBds0pjBqAlkd25HN7rOrFleaJ1/
→ctaJxQZBKT5ZPt0m9STJEadao0xAH0ahmbWn\nn01FuhjuefXKNEgV4We0+UXgVCwOPjdAvBbI+e0ocS3MFEVzG6uBQE3xDk3Szy
→rOPNk3sgrDQoo//fb4hVC1CLQJ13hef4Y53CI\nnrU7m2Ys6xt0nUW7/
→vGT1M0NPAGMBAAGjQjBAMA4GA1UdDwEB/wQEAwIBBjAPBgNV\nnHRMBAf8EBTADAQH/
→MB0GA1UdDgQWBBER5tFnme7b15AFzgAiIyBpY9umbbjANBgkq\nnhkiG9w0BAQsFAAOCAgEAVR9YqbyyqFDQDLHYGmkGjYkIrGF1
→V91ZL\nnubhzEFnTIZd+50xx+7LSYK05qAvqFyFWhfQDlnrzuBZ6brJFe+GnY+EgPbk6ZGQ\nn3BebYhtF8GaV0nxvwuo77x/
→Py9auJ/GpsMiu/X1+mvoiBOv/2X/qkSsisRcOj/
→KK\nnNftY2PwByVS5UcBmiogziUwthDyC3+6WVwW6LLv3xLfHTjuCvjHIInNzktHCgKQ5\nnORAZI4JMPJ+GslWYHb4phowim57ia
→+sKAIuvtd7u+Nxe5AW0wdeRlN8NwdC\nnjNPelpzVmbUq4JUagEiuTDkHszxHpFKVK7q4+63SM1N95R1NbdWhscdCb+ZAJzVc\nn
→1lvh+wjChP4kqK0J2qxq\nn4RgqsahDYVvTh9w7jXbyLeiNdd8XM2w9U/t7y0Fff/
→9yi0GE44Za4rF2LN9d11TPA\nnmRGunUHBcnWEvgJBQl9nJEiU0Zsnvgc/
→ubhPgXRR4Xq37Z0j4r7glSgEEzwxA57d\nnemyPxcgYxn/eR44/KJ4EBs+lVDR3veyJm+kXQ99b21/
→jh5Xos1AnX5iItreGCC=\nn-----END CERTIFICATE-----"
],
"wap_onion_url": "",
"wap_url": "ws://localhost:8080/v2/ws"
},
"mainnet": {
"address_explorer_url": "https://blockstream.info/address/",
"bech32_prefix": "bc",
"bip21_prefix": "bitcoin",
"blob_server_onion_url": "",
"blob_server_url": "",
"csv_buckets": [
25920,

```

(continues on next page)

(continued from previous page)

```

51840,
65535
],
"development": false,
"electrum_onion_url": "explorerzydxu5ecjrkwceayqybizmpjjznk5izmitf2modhcsuqlid.
→ onion:110",
"electrum_tls": true,
"electrum_url": "blockstream.info:700",
"liquid": false,
"mainnet": true,
"max_reorg_blocks": 144,
"name": "Bitcoin",
"network": "mainnet",
"p2pkh_version": 0,
"p2sh_version": 5,
"pin_server_onion_url": "http://
→ mrrxtq6tjpbnbm7vh5jt6mpjctn7ggyfy5wegvbef3x7jrznqawlmid.onion",
"pin_server_public_key":
→ "0332b7b1348bde8ca4b46b9dcc30320e140ca26428160a27bdbfc30b34ec87c547",
"pin_server_url": "https://jade.pin.blockstream.com",
"price_onion_url": "http://
→ greenv32e5p4rax6dmfqb4zzl7kq2fbmizd7miyava2actplmipyx2qd.onion/prices",
"price_url": "https://green-bitcoin-mainnet.blockstream.com/prices",
"server_type": "green",
"service_chain_code":
→ "e9a563d68686999af372a33157209c6860fe79197a4dafd9ec1dbaa49523351d",
"service_pubkey":
→ "0322c5f5c9c4b9d1c3e22ca995e200d724c2d7d8b6953f7b38fddf9296053c961f",
"spv_enabled": false,
"spv_multi": false,
"spv_servers": [],
"tx_explorer_url": "https://blockstream.info/tx/",
"wamp_cert_pins": [
    "46494e30379059df18be52124305e606fc59070e5b21076ce113954b60517cda",
    "b42688d73bac5099d9cf4fdb7b05f5e54e98c5aa8ab56ee06c297a9a84d2d5f1",
    "67add1166b020ae61b8f5fc96813c04c2aa589960796865572a3c7e737613dfd",
    "64e286b76063602a372efd60cde8db2656a49ee15e84254b3d6eb5fe38f4288b"
],
"wamp_cert_roots": [
    "\n-----BEGIN CERTIFICATE-----\nMIICxjCCAk2gAwIBAgIRAL093/
→ inhFu86QOgQTWzSkUwCgYIKoZIzj0EAwMwTzEL\nMAkGA1UEBhMCVVMxKTAnBgNVBAoTIEludGVybmV0IFNlY3VyaXR5IFJlc2
→ xwLCA/TmC9N/A8AmzfwdAVXMpcuBe8qQyWj\n+240JxP2T35p0wKZXuskR5LBjJvmsSGPwSSB/
→ GjMH2m6WPUZlvd0xhagggEIMIIb\nBDAOBgNVHQ8BAf8EBAMCAYYwHQYDVR0lBBYwFAYIKwYBBQUHAWIGCCsGAQUFBwMB\n+nmBI
→ wQIMAYBAf8CAQAwHQYDVR0OBBYEFFrz7Sv8NsI3eblSMOpUb89V\nnyy6sMB8GA1UdIwQYMBaAFHxClq7eS0g7+pL4nozPbYupc
→ AwCd89eYhNlXlr3pD5xcSAQH8suzYHK0l9YST8pE9kLJ03uGqQIwWrGxtO3q\nyYJkgsTgDyJ2gJrjubi1K9sZmHzOa25JK1fUpl
→ Lgul/\n-----END CERTIFICATE-----",
    "\n-----BEGIN CERTIFICATE-----\nMIICxjCCAkYgAwIBAgIQTtI99q9+x/
→ mwXHJv+VEqdzAKBggqhkJOPQDAzBPMQsw\nCQYDVQQGEwJVUzEpMCcGA1UEChMgSW50ZXJuZXQgU2VjdXJpdHkgUmVzZWZyY2
→ LbBVDxCSnsI7vuf4xp4Cu+ETnggGE\nnyBqJ3j8iUwe5Yt/qfSrRf1/
→ D5R58duaJ+IvLRXeASRqEL+VkdXrW3qOCAQgwggEE\nnMA4GA1UdDwEB/
→ wQEAWIBhjAdBgNVHSUEFjAUBgggrBgEFBQcDAgYIKwYBBQUHAWew\nnEgYDVR0TAQH/BAGwBgEB/
→ wIBADAdBgNVHQ4EFgQUbZkq9U0C6+MRwWC6km+NPS7x\nn6kQwHwYDVR0jBBgwFoAUFEKWr5LSDv6kviejm9t161yN5UwMgYIK
→ No+KQ\nnUAM1u34E36neL/Zba7ombkIOChSgx1iVxztqFWGddgoG+tpRPWhuhhn\n-----END
→ CERTIFICATE-----",
    "\n-----BEGIN CERTIFICATE-----
→ \nMIIFWjCCAOKgAwIBAgIQbkepxUtHDA3sM9CJuRz04TANBgkqhkiG9w0BAQwFADBh\nnMQswCQYDVQQGEwJVUzEiMCAgA1UEChM
→ vo27xqLpeXo4xL+SV2sfnOhB2x+cWX3u+58qPpvBKJXqeqUqv4IyflpLGcY9vX\nnmX7wCl7raKb0xlpHdu0QM+NosROjyBhsS+
→ SCYe7\nnzUjwTcLceoiKu7rPWRnWr4+wB7CeMfGCwDfLqZtbBkOtdh+JhpFAz2weaSUKK0
→ kB+Xj9e1x3+naH+uzfsQ551Ve0vSbv1gHR6xYKu44LtcXFilWr06zqkUsp\nnzBmkMiVOKvFLRNACzqrOSbTqn3yDsEB750Orp2
→ VjsPOS+C12LOO\nnRc92wOIAK/
→ LDjCnRISnsqIA94xlcx36m97PtbfkSIS5r762DL8EGMUUXLeXdYw\nnk70paDPvOmbSB4om3xPXV2V4J95eSRQAogB/
→ mgqghtqmxlbCluQ0WEdrhEbEg8QOB+\nndVrNVjzrlw5y0vtOUucxD/SVRNuJLDWcfr0wbrM7Rv1/
→ oFB2ACYPTIrInqYnXgF\nnlQIDAQAB0IwQDAOBGNVHQ8BAf8EBAMCAQYwDwYDVR0TAQH/BAUwAwEB/
→ zAdBgNV\nnHQ4EFgQU5K8rJnEaK0gnhS9SZizv8IkTcT4wDQYJKoZIhvcNAQEMBQADggIBADiW\nnC49tJYeX++dnAsznyvgv3
→ -----END-----"

```

(continued from previous page)

```

"\n-----BEGIN CERTIFICATE-----
↪\nMIIFazCCA1OgAwIBAgIRAIQz7DSQONZRGPGu2OCiwAwDQYJKoZIhvcNAQELBQAw\nTzELMAkGA1UEBhMCVVMxKTAnBgNVBA
↪vVqbvYATyjb3miGbESTtrFj/RQSa78f0uoxmyF+\n0TM8ukj13Xnfs7j/
↪EvEhmkvBioZxaUpmZmyPfjxwv60pIgbz5MDmgK7iS4+3mX6U\nnA5/
↪TR5d8mUgjU+g4rk8Kb4Mu0U1XjIB0ttov0DiNewNwIRt18jA8+o+u3dpjq+sW\nnT8KOEut+zwvo/
↪7V3LvSye0rgTBI1DHCNAymg4VMk7BPZ7hm/
↪ELNKjD+Jo2FR3qyH\nb5T0Y3HsLuJvW5iB4Y1cNHlsdu87kGJ55tukmi8mxdAQ4Q7e2RCOFvu396j3x+UC\nb5iPNgiV5+I31g
↪1JBdiB3QW0KtZB6awBdpUKD9jflb0SHzUv\nkBds0pjBqAlkd25HN7rOrFleaJ1/
↪ctaJxQZBKT5ZPt0m9STJEadao0xAH0ahmbWn\n0lFuhjuefXKnEgV4We0+UXgVCwOPjdAvBbI+e0ocS3MFEVzG6uBQE3xDk3Sz
↪rOPNk3sgrDQoo//fb4hVC1CLQJ13hef4Y53CI\nnrU7m2Ys6xt0nUW7/
↪vGT1M0NPAgMBAAGjQjBAMA4GA1UdWEB/wQEAWIBBjAPBgNV\nnHRMBAf8EBTADAQH/
↪MB0GA1UdDgQWBRR5tFnme7b15AFzgAiIyBpY9umbbjANBgkq\nnhkiG9w0BAQsFAAOCAGEAVR9YqbyyqFDQDLHYGmkGJykIrGF1
↪V91ZL\nubhzEFnTIZd+50xx+7LSYK05qAvqFyFWhfQDlnrzuBZ6brJFe+GnY+EgPbk6ZGQ\nn3BebYhtF8GaV0nxvwuo77x/
↪Py9auJ/GpsMiu/X1+mvoiBOv/2X/qkSsisRcOj/
↪KK\nnNftY2PwByVS5uCbMiogziUwthDyC3+6WVwW6LLv3xLfHTjuCvjHIInNzktHCgKQ5\nnORAZI4JMPJ+GslWYHb4phowim57ia
↪+sKAiuvtd7u+Nxe5AW0wdeR1N8NwdC\nnjNPElpzVmbUq4JUagEiuTDkHszxHpFKVK7q4+63SM1N95R1NbdWhscdCb+ZAJzVc\nn
↪1lvh+wjChP4kqK0J2qxq\nn4RgqsahDYVvTH9w7jXbyLeiNdd8XM2w9U/t7y0Ff/
↪9yi0GE44Za4rF2LN9d11TPA\nnmRGunUHBcnWEvgJBQ19nJEiU0Zsnvgc/
↪ubhPgXRR4Xq37Z0j4r7g1SgEEzwxAS7d\nnemyPxgcYxn/eR44/KJ4EBs+lVDR3veyJm+kXQ99b21/
↪+jh5Xos1AnX5iItreGCc=\n-----END CERTIFICATE-----"
    ],
    "wamp_onion_url": "ws://greenv32e5p4rax6dmfgeb4zzl7kq2fbmizd7miyava2actplmipyx2qd.
↪onion:80/v2/ws",
    "wamp_url": "wss://green-bitcoin-mainnet.blockstream.com/v2/ws"
  },
  "testnet": {
    "address_explorer_url": "https://blockstream.info/testnet/address/",
    "bech32_prefix": "tb",
    "bip21_prefix": "bitcoin",
    "blob_server_onion_url": "",
    "blob_server_url": "",
    "csv_buckets": [
      144,
      4320,
      51840
    ],
    "development": false,
    "electrum_onion_url": "explorerzydxu5ecjrkwceayqybizmpjjznk5izmitf2modhcsuqlid.
↪onion:143",
    "electrum_tls": true,
    "electrum_url": "blockstream.info:993",
    "liquid": false,
    "mainnet": false,
    "max_reorg_blocks": 1008,
    "name": "Testnet",
    "network": "testnet",
    "p2pkh_version": 111,
    "p2sh_version": 196,
    "pin_server_onion_url": "http://
↪mrrxtq6tjpbnbm7vh5jt6mpjctn7ggyfy5wegvbef3x7jrznqawlmid.onion",
    "pin_server_public_key":
↪"0332b7b1348bde8ca4b46b9dcc30320e140ca26428160a27bdbfc30b34ec87c547",
    "pin_server_url": "https://jadepin.blockstream.com",
    "price_onion_url": "http://
↪greent5yfxruca52pkqjtgo2qdxijscqlastnv3jwzpmavvffldm2yd.onion/prices",
    "price_url": "https://green-bitcoin-testnet.blockstream.com/prices",
    "server_type": "green",

```

(continues on next page)

(continued from previous page)

```

"service_chain_code":
→ "b60befcc619bb1c212732770fe181f2f1aa824ab89f8aabb49f2e13e3a56f0f04",
"service_pubkey":
→ "036307e560072ed6ce0aa5465534fb5c258a2ccfbcb257f369e8e7a181b16d897b3",
"spv_enabled": false,
"spv_multi": false,
"spv_servers": [],
"tx_explorer_url": "https://blockstream.info/testnet/tx/",
"wamp_cert_pins": [],
"wamp_cert_roots": [
    "\n-----BEGIN CERTIFICATE-----\nMIICXjCCAk2gAwIBAgIRALO93/
→ inhFu86QOgQTWzSkUwCgYIKoZIzj0EAwMwTzEL\nMAKGA1UEBhMCVVMxKTAnBgNVBAoTIEludGVybmV0IFNlY3VyaXR5IFJlc2V
→ xwLCA/TmC9N/A8AmzfwdAVXMpcuBe8qQyWj\n+240JxP2T35p0wKZXusKR5LBJJvmsSGPwSSB/
→ GjMH2m6WPUZIVd0xhajggEIMIIB\nBDAOBgNVHQ8BAf8EBAMCAYYwHQYDVR01BBYwFAYIKwYBBQUHAWIGCCsGAQUFBwMB\nnMBI
→ wQIMAYBAf8CAQAwHQYDVR0OBBYEFFrz7Sv8NsI3eblSMOpUb89V\nnyy6sMB8GA1UdIwQYMBaAFHxClq7eS0g7+pL4nozPbYupc
→ AwCd89eYhNlXLR3pD5xcSAQh8suzYHK019YST8pE9kLJ03uGgQIwWrGxtO3q\nnYJkgsTgDyj2gJrjubilK9sZmHzOa25JK1fUpl
→ Lgul/\n-----END CERTIFICATE-----",
    "\n-----BEGIN CERTIFICATE-----\nMIICXjCCAkYgAwIBAgIQTtI99q9+x/
→ mwXHJv+VEqdzAKBggqhkJOPQDQAzBPMQsw\nnCQYDVQQGEwJVUzEpMCCGA1UEChMgSW50ZXJuZXQgU2VjdXJpdHkgUmVzZWYy
→ LbBVDXcnsI7vuf4xp4Cu+ETNggGE\nnyBqJ3j8iUwe5Yt/qfSrRf1/
→ D5R58duaJ+IvLRXeASRqEL+VkdXrW3qOCAQgwgwEE\nnMA4GA1UdDwEB/
→ wQEAwIBhjAdBgNVHSUEFjAUBgggrBgEFBQcDAgYIKwYBBQUHAWew\nnEgYDVR0TAQH/BAgwBgEB/
→ wIBADAdBgNVHQ4EFgQUbZkq9U0C6+MRwWC6km+NPS7x\nn6kQwHwYDVR0jBBGwFoAUfEKWrt5LSdv6kviejm9ti6lyN5UwMgYIKv
→ No+KQ\nnUAM1u34E36neL/Zba7ombkIOchSgx1iVxztqFWGddgoG+tpPRPWhuhhn\nn-----END
→ CERTIFICATE-----",
    "\n-----BEGIN CERTIFICATE-----
→ \nMIIFWjCCAOKgAwIBAgIQbkepxUtHDA3sM9CJuRz04TANBgkqhkiG9w0BAQwFADBH\nnMQswCQYDVQQGEwJVUzEiMCAGA1UEChM
→ vo27xqLpeXo4xL+Sv2sfNohB2x+cWX3u+58qPpvBKJXqeQUqv4IyflpLGcy9vX\nnmX7wCl7raKb0xlpHdu0QM+NOsROjYbhsS+
→ SCYe7\nnzUjwTcLCoiku7PWRnWr4+wB7CeMfGCwDfLqZtbBkOtdh+JhpFAz2weaSUKKOP\nnfylblqAj+lug8aJRT7oM6iCsVl
→ kB+Xj9e1x3+naH+uzfsQ551Ve0vSbvlGHR6xYku44LtcXFilWr06zqkUsp\nnzBmkMiVOKvFfLRNACzqrOSbTqn3yDsEB750Orp2y
→ VjsPOS+C12LOO\nnRc92wOIAK/
→ 1TD7Cn1TsNsYqiA94xrcx36m97PtbfkSIS5r762DL8EGMUUXLeXdYw\nnk70paDPvOmsB4om3xPXV2V4J95eSRQAogB/
→ mqghtqmxlbCluQ0WEdrHbEg8QOB+\nndVrNVjzRlw5y0vtOUucXD/SVRNuJLDWcfr0wbrM7Rv1/
→ oFB2ACYPTrIrnqYNxgF\nnlQIDAQABO0IwQDAOBGNVHQ8BAf8EBAMCAQYwDwYDVR0TAQH/BAUwAwEB/
→ zAdBgNV\nnHQ4EFgQU5K8rJnEaK0gnhS9Sizv8IkTcT4wDQYJKoZIhvcNAQEMBQADgIBADiW\nnCu49tJYeX+dnAsznyvgvy3
→ UEBZvXw6Z\nnXPYfcX3v73svfu02lpdwCxxU11xWaj0140k4DLh9+42FpLFZXvRq4d2h9mREruZR\nngyFmxhE+885H7pwoHyXa/
→ 6xmld01D1zvICxi/ZG6qcz8WpyTgYmpl0p8WnK00dC3\nnd8t5/
→ Wk6kjftbjhlRn7pYL15iJdfOBL07q9bgsiG1eGZbYwE8na6SfZu6W0eX6Dv\nnJ4J2QPim01hcDyxC2kLGe4g0x8HYRZvBPsVhH
→ Zg\nnDdIX02OYI5NaAIFiTo/
→ Nis3Jz5nu2Z6qNuFoS3FJFDYoOj0dzpqPJeaAcWErtXvM\nn+SUWgeExX6GjfhaknBZqlxi9dnKlC54dNuYvoS+cJEPqOba+MS
→ ZRBI1s6Kl0obsXs7X9\nnSQ98POyDGCBDTtWTurQ0sR8WNh8M5mQ5Fkzc4P4dyKliPUdQysU0ArSuiYgzNdws\nnE3PYJ/
→ HQcu51OyLemGhmW/HGY0dVHLq1CFF1pkgl\nn-----END CERTIFICATE-----",
    "\n-----BEGIN CERTIFICATE-----
→ \nMIIFazCCA1OgAwIBAgIRAIQz7DSQONZRGpGu2OCiwAwDQYJKoZIhvcNAQELBQAw\nnTzELMAKGA1UEBhMCVVMxKTAnBgNVBAoTIEludGVybmV0IFNlY3VyaXR5IFJlc2V
→ vVqbvYATyjb3miGbESTtrFj/RQSa78f0uoxmyF+\nn0TM8ukj13Xnfs7j/
→ EvEhmKvBioZxaUpmZmyPfxwv60pIgbz5MDmgK7iS4+3mX6U\nnA5/
→ TR5d8mUgJU+g4rk8Kb4Mu0U1XjIB0ttov0DiNewNIRt18jA8+o+u3dpjq+sW\nnT8KOEut+zwvo/
→ 7V3LvSye0rgTBIldHCNAymg4VMk7BPZ7hm/
→ ELNKjD+Jo2FR3qyH\nnB5T0Y3HsLuJvW5iB4Y1cNHlsdu87kGJ55tukmi8mxdAQ4Q7e2RCOFvu396j3x+UC\nnB5iPNgiV5+I3lq
→ 1JBdiB3QW0ktZB6awBdpUKD9jflb0SHzUv\nnKBds0pjBqAlkd25HN7rOrFleaJ1/
→ ctaJxQZBKT5ZPt0m9STJEadao0xAH0ahmbWn\nnOlFuhjuefXKNEgV4We0+UXgVCwOpjdAvBbI+e0ocS3MFEVzG6uBQE3xDk3Szy
→ rOPNk3sgrDQoo//fb4hVC1CLQJ13hef4Y53CI\nnrU7m2Ys6xt0nUW7/
→ vGT1M0NPagMBAAGjQjBAMA4GA1UdDwEB/wQEAwIBBjAPBgNV\nnHRMBAf8EBTADAQH/
→ MB0GA1UdDgQWBRR5tFnme7b15AFzgAiIyBpY9umbbjANBgkq\nnhkiG9w0BAQsFAAOCAgEAVR9YqbyyqFDQDLHYGmkGjYkIrGF1
→ V91ZL\nnubhzEfNTIzd+50xx+7LSYK05qAvqFyFWhfQDlnrzuBZ6brJFe+GnY+EGPbk6ZGQ\nn3BebYhtF8GaV0nxvwuo77x/
→ Py9auJ/GpsMiu/X1+mvoiBOv/2X/qkSsisRcOj/
→ KK\nnNFtY2PwByVS5uCbMiogziUwthDyC3+6WVw6LLv3xLfHTjuCvjHIInNzktHCgKQ5\nnORAZi4JMPJ+GslWYHb4phowim57ia
→ +sKAIuvtd7u+Nxe5AW0wdeRLN8NwdC\nnjNPElpzVmbUq4JUagEiUTDkHszxHpFKVK7q4+63CmHn5mHxZuBcdCb+ZAJzVc\nn0
→ 1lvh+wjChP4kqKOJ2qxq\nn4RgqsahDYvTH9w7jXbyLeiNdd8XM2w9U/t7y0Ff/
→ 9yi0GE44Za4rF2LN9d11TPA\nnmRGunUHBcnWEvgJBQI9nJEiU0Zsnvgc/
→ ubnHGAkK4xq720j417g1SgEEzwa57d\nnemyPxcgYxn/er44/KJ4EBs+1VDR3veyJm+kXQ99b21/
→ +jh5Xos1AnX5iItreGCC=\nn-----END CERTIFICATE-----"

```

(continues on next page)

(continued from previous page)

```

    ],
    "wamp_onion_url": "ws://greent5yfxruca52pkqjtgo2qdxijscqlastnv3jwzpmavvffdldm2yd.
↪onion:80/v2/ws",
    "wamp_url": "wss://green-bitcoin-testnet.blockstream.com/v2/ws"
  },
  "testnet-liquid": {
    "address_explorer_url": "https://explora.blockstream.com/liquidtestnet/address/",
    "asset_registry_onion_url": "http://
↪lhquhzzpzg5tyymcqep24fynpzzqgg3m3rlh7ascnw5cpqsro35bfxyd.onion/testnet/",
    "asset_registry_url": "https://assets-testnet.blockstream.info/",
    "bech32_prefix": "tex",
    "bip21_prefix": "liquidtestnet",
    "blech32_prefix": "tlq",
    "blinded_prefix": 23,
    "blob_server_onion_url": "",
    "blob_server_url": "",
    "csv_buckets": [
      1440,
      65535
    ],
    "development": false,
    "electrum_onion_url": "explorerzydxu5ecjrkwcayqybizmpjjznk5izmitf2modhcsuqlid.
↪onion:587",
    "electrum_tls": true,
    "electrum_url": "blockstream.info:465",
    "liquid": true,
    "mainnet": false,
    "max_reorg_blocks": 2,
    "name": "Testnet Liquid",
    "network": "testnet-liquid",
    "p2pkh_version": 36,
    "p2sh_version": 19,
    "pin_server_onion_url": "http://
↪mrrxtq6tjpbnbm7vh5jt6mpjctn7ggyfy5wegvbef3x7jrznqawlmid.onion",
    "pin_server_public_key":
↪"0332b7b1348bde8ca4b46b9dcc30320e140ca26428160a27bdbfc30b34ec87c547",
    "pin_server_url": "https://jadepin.blockstream.com",
    "policy_asset": "144c654344aa716d6f3abcc1ca90e5641e4e2a7f633bc09fe3baf64585819a49
↪",
    "price_onion_url": "http://
↪greent5yfxruca52pkqjtgo2qdxijscqlastnv3jwzpmavvffdldm2yd.onion/prices",
    "price_url": "https://green-bitcoin-testnet.blockstream.com/prices",
    "server_type": "green",
    "service_chain_code":
↪"c660eec6d9c536f4121854146da22e02d4c91d72af004d41729b9a592f0788e5",
    "service_pubkey":
↪"02c47d84a5b256ee3c29df89642d14b6ed73d17a2b8af0aca18f6f1900f1633533",
    "spv_enabled": false,
    "spv_multi": false,
    "spv_servers": [],
    "tx_explorer_url": "https://explora.blockstream.com/liquidtestnet/tx/",
    "wamp_cert_pins": [],
    "wamp_cert_roots": [
      "\n-----BEGIN CERTIFICATE-----\nMIICXjCCAk2gAwIBAgIRAL093/
↪inhFu86QOgQTWzSkUwCgYIKoZIZj0EAWMwTzEL\nMAkGAlUEBhMCMVVMxKTAnBgNVBAoTIEludGVybmV0IFN1Y3VyaXR5IFJlc2V
↪xwLCA/TmC9N/A8AmzfwdAVXMpcuBe8qQyWj\n+240JxP2T35p0wKZXuskR5LBJJvmsSGPwSSB/
↪GjMH2m6WPUZIVd0xhajggEIMIIB\nBDAOBgNVHQ8BAf8EBAMCAYYwHQYDVR0lBBYwFAYIKwYBBQUHAWIGCCsGAQUFBwMB\nnMBIC
↪wQTMAYBAf8CAQAwHQYDVR0OBBYEFFrz7Sv8NsI3eb1SMOpUb89V\nnyy6sMB8GA1UdIwQYMBI
↪AwCd89eYhN1XLR3pD5xcSAQh8suzYHK019YST8pE9kLJ03uGqQIWwRgxtO3q\nnyYJkgsTgDyJ2gJrjubi1K9sZmHzOa25JK1fUp
↪Lgu1\n-----END CERTIFICATE-----",

```

(continues on next page)

(continued from previous page)

```

"\n-----BEGIN CERTIFICATE-----\nMIICxjCCAkYgAwIBAgIQTtI99q9+x/
→mwXHJv+VEqdzAKBggqhkJOPQDZBPMQsw\nCQYDVQQGEwJVUzEpMCcGA1UEChMgSW50ZXJuZXQgU2VjdXJpdHkgUmVzZWZyY2
→LbBVDXcnsI7vuf4xp4Cu+ETNggGE\nyBqJ3j8iUwe5Yt/qfSrRf1/
→D5R58duaJ+IvLRXeASRqEL+VkdXrW3qOCAQgwgGE\nnMA4GA1UdDwEB/
→wQEAwIBhjAdBgNVHSUEFjAUBgggrBgEFBQcDAGYIKWYBBQUHAAwEw\nnEgYDVROTAQH/BAGwBgEB/
→wIBADAdBgNVHQ4EFgQUbZkq9U0C6+MRwWC6km+NPS7x\nn6kQwHwYDVR0jBBGwFoAUfEKWrt5LSdv6kviejm9t161yN5UwMgYIK
→No+KQ\nnUAM1u34E36neL/Zba7ombkIOchSgx1iVxztqFWGddgoG+tpRPWhuhhn\nn-----END
→CERTIFICATE-----",

"\n-----BEGIN CERTIFICATE-----
→\nMIIFWjCCA0KgAwIBAgIQbkepxUtHDA3sM9CJuRz04TANBgqhkiG9w0BAQwFADBH\nnMQswCQYDVQQGEwJVUzEiMCAGA1UEChM
→vo27xqLpeXo4xL+Sv2sfnOhB2x+cWX3u+58qPpvBKJXqeqUqv4IyflPLGcY9vX\nnmX7wC17raKb0xlpHdU0QM+NOSROjyBhsS+
→SCYe7\nnzUjwTcLceoiKu7rPWRnWr4+wB7CeMfGCwcDfLqZtbBkOtdh+JhpFAz2weaSUKKOP\nnfyblqAj+lug8aJRT7oM6iCsV1
→kB+Xj9e1x3+naH+uzfsQ551Ve0vSbvlgHR6xYKu44LtcXFilWr06zqkUsp\nnzBmkMiVOKvFlRNACzqrOSbTqn3yDsEB750Orp2
→VjsPOS+C12LOO\nnRc92wO1AK/
→1TD7Cn1TsNsYqiA94xrcx36m97PtbfkSIS5r762DL8EGMUUXLeXdYw\nnk70paDPvOmsB4om3xPXV2V4J95eSRQAogB/
→mqgghtqmxlbCluQ0WEdrHbEg8QOB+\nndVrNVjzRlw5y0vtOUucD/SVRNuJLDWcfr0wbrM7Rv1/
→oFB2ACYPTTrIrnqYNxgF\nnlQIDAQABO0IwQDAOBGNVHQ8BAF8EBAMCAQYwDwYDVR0TAQH/BAUwAwEB/
→zAdBgNV\nnHQ4EFgQU5K8rJnEaK0gnhS9S2izv8IkTcT4wDQYJKoZIhvcNAQEMBAQDggIBADiW\nnCu49tJYeX++dnAsznyvgv3
→UEBZvXw6Z\nnXPYfcX3v73svfu02lpdwCxu11xWajol40k4DLh9+42FpLFZxvRq4d2h9mREruZR\nngyFmxhE+885H7pwoHyXa/
→6xmld01D1zvICxi/ZG6qcZ8WpyTgYmpl0p8WnK0OdC3\nnd8t5/
→Wk6kjftbjhlRn7pYL15iJdfOBL07q9bgsiG1eGZbYwE8na6SfZu6W0eX6Dv\nnJ4J2QPim01hcDyx2kLGe4g0x8HYRZvBPsVhH
→Zg\nnDdIx02OYI5NaAIFiTO/
→Nis3Jz5nu2Z6qNuFoS3FJFYDoOj0dzpqPJeaAcWErtXvM\nn+SUWgeExX6GjfhaknBZqlxi9dnK1C54dNuYvoS++cJEPqOba+MS
→ZRBiIs6Kl0obsXs7X9\nnSQ98POyDGCBDTtWTurQ0sR8WNh8M5mq5Fkzc4P4dyKliPUDqysU0ArSuiYgzNdws\nnE3PYJ/
→HQcu51OyLemGhmW/HGY0dVHLqlCFF1pkgl\nn-----END CERTIFICATE-----",

"\n-----BEGIN CERTIFICATE-----
→\nMIIFazCCA0GwAwIBAgIRAIQz7DSQONZRGpGu2OCiwAwDQYJKoZIhvcNAQELBQAw\nnTzELMAkGA1UEBhMCVVMxKTAnBgNVBA
→vVqbvYATyjb3miGbESTtrFj/RQSa78f0uoxmyF+\nnoTM8ukj13Xnfs7j/
→EvEhmkvBioZxaUpmZmyPfjxwv60pIgbz5MDmgK7iS4+3mX6U\nnA5/
→TR5d8mUgJ+g4rk8Kb4Mu0U1XjIB0ttov0DiNewNwIRt18jA8+o+u3dpjq+sW\nnT8KOEut+zwvo/
→7V3LvSye0rgTBIldHCNAymg4VMk7BPZ7hm/
→ELNKjD+Jo2FR3qyH\nnB5T0Y3HSLuJvW5iB4Y1cNHlsdu87kGJ55tukmi8mxdAQ4Q7e2RCOFvu396j3x+UC\nnB5iPNgiV5+I3lg
→1JBdiB3QW0KtZB6awBdpUKD9jflb0SHzUv\nnKBds0pjBqAlkd25HN7rOrFleaJ1/
→ctaJxQZBKT5ZPt0m9STJEadao0xAH0ahmbWn\nnO1FuhjuefXKNEgV4We0+UXgVCwOPjdAvBbI+e0ocS3MFEVzG6uBQE3xDk3Sz
→rOPNk3sgrDQoo//fb4hVC1CLQJ13hef4Y53CI\nnrU7m2Ys6xt0nUW7/
→vGT1M0NPAGMBAAGjQjBAMA4GA1UdDwEB/wQEAwIBBjAPBgNV\nnHRMBAF8EBTADAQH/
→MB0GA1UdDgQWBRR5tFnme7b15AFzgAiIyBpY9umbbjANBgkq\nnhkiG9w0BAQsFAAOCAgEAVR9YqbyyqFDQDLHYGmkGJykIrGF1
→V91ZL\nnubhzEFnTIZd+50xx+7LSYK05qAvqFyFWhfQDlnrzuBZ6brJFe+GnY+EgPbk6ZGQ\nn3BebYhtF8GaV0nxvwuo77x/
→Py9auJ/GpsMiu/X1+mvoiBOv/2X/qkSsisRcOj/
→KK\nnNFtY2PwByVS5uCbMioGziUwthDyC3+6WVwW6LLv3xLfHTjuCvjHIInNzktHCgKQ5\nnORAZI4JMPJ+GslWYHb4phowim57ia
→+sKAIuvtd7u+Nxe5AW0wdeRlN8NwdC\nnjNPElpzVmbUq4JUagEiuTDkHszxHpFKVK7q4+63SM1N95R1NbdWhscdCb+ZAJzVc\nn
→1lvh+wjChP4kqK0J2qxq\nn4RgqsahDYVvTH9w7jXbyLeiNdd8XM2w9U/t7y0Fff/
→9yi0GE44Za4rF2LN9d11TPA\nnmRGunUHBcnWEvgJBQ19nJEiU0Zsnvgc/
→ubhPgXRR4Xq37Z0j4r7g1SgEEzwxAS7d\nnemyPxcgYxn/eR44/KJ4EBs+1VDR3veyJm+kXQ99b21/
→+jh5Xos1AnX5iItreGCC=\nn-----END CERTIFICATE-----"
1,
  "wamp_onion_url": "ws://liqtestulh46kwa3mgenugrcogvjvzr2qdto663hujwnbaewzpkoad.
→onion/v2/ws",
  "wamp_url": "wss://green-liquid-testnet.blockstream.com/v2/ws"
}

```

2.49 Transaction limits JSON

```
{ "is_fiat": false, "mbtc": "555" }
{ "is_fiat": true, "fiat": "555" }
```

2.50 Two-factor detail JSON

```
{ "confirmed": true, "data": "mail@example.com", "enabled": true }
```

2.51 Auth handler status JSON

Describes the status of a `GA_auth_handler`. Returned by `GA_auth_handler_get_status`.

All status JSON contains a `"name"` element with the name of the handler being invoked.

The remaining data returned depends on the current state of the handler, as follows:

- `"done"`:

```
{
  "status": "done",
  "action": "disable_2fa",
  "result": {}
}
```

action The action being processed.

result The data returned from the call, if any.

- `"error"`:

```
{
  "status": "error",
  "action": "disable_2fa",
  "error": "Incorrect code"
}
```

action The action being processed.

error A text description of the error that occurred.

- `"call"`:

```
{
  "status": "call",
  "action": "disable_2fa"
}
```

action The action being processed.

- `"request_code"`:


```
{
  "status": "request_code",
  "action": "disable_2fa",
  "methods": [ "email", "sms", "phone", "gauth", "telegram" ]
}
```

action The action being processed.

methods A list of available two factor methods the user has enabled, or the single element "data" if the call requires more data to continue.

- "resolve_code" (two factor):

```
{
  "status": "resolve_code",
  "action": "disable_2fa",
  "method": "email",
  "auth_data": {},
  "attempts_remaining": "3"
}
```

action The action being processed.

method The two factor method the user should fetch the code to enter from.

auth_data Method-specific ancillary data for resolving the call.

attempts_remaining If present, the number of incorrect attempts that can be made before the call fails.

- "resolve_code" (hardware wallet/external device):

```
{
  "status": "resolve_code",
  "action": "disable_2fa",
  "required_data": {
    "action": "get_xpubs",
    "device": {}
  }
}
```

action The action being processed.

required_data Contains the data the HWW must provide, see *GDK Hardware Wallet Interface*.

- "resolve_code" (request for additional data):

```
{
  "status": "resolve_code",
  "action": "data",
  "method": "data",
  "auth_data": {}
}
```

action Always "data".

method Always "data".

auth_data Method-specific ancillary data for processing the additional data request.

2.52 Reconnect JSON

Controls session and internal Tor instance reconnection behavior.

```
{
  "hint": "connect",
  "tor_hint": "connect"
}
```

hint Optional, must be either "connect" or "disconnect" if given.

tor_hint Optional, must be either "connect" or "disconnect" if given.

For both hint types, "disconnect" will disconnect the underlying network connection used by the session, while "connect" will reconnect it. If a hint is not given, no action will be taken for that connection type.

Each session will automatically attempt to reconnect in the background when they detect a disconnection, unless "disconnect" is passed to close the connection first. The session will be notified using a *Network notification* when the underlying network connection changes state.

For environments such as mobile devices where networking may become unavailable to the callers application, the network must be disconnected and reconnected using *GA_reconnect_hint* in order for connectivity to be resumed successfully. In particular, when using the built-in Tor implementation to connect, failure to do so may result in Tor failing to connect for the remaining lifetime of the application (this is a Tor limitation).

2.53 Convert amount JSON

Amounts to convert are passed with a single key containing the unit value to convert, returning all possible conversion values for that value. See *Amount JSON* for the list of unit values available.

For example, to convert satoshi into all available units:

```
{
  "satoshi": 1120
}
```

If "fiat_currency" and "fiat_rate" members are provided, the fiat conversion will fall back on these values if no fiat rates are available. Callers can check the "is_current" member in the result *Amount JSON* to determine if the fall back values were used.

For example, to convert bits into all available units, with a fiat conversion fallback:

```
{
  "bits": "20344.69",
  "fiat_currency": "USD",
  "fiat_rate": "42161.22"
}
```

2.54 Amount JSON

```
{
  "bits": "20344.69",
  "btc": "0.02034469",

```

(continues on next page)

(continued from previous page)

```

"fiat": "0.02",
"fiat_currency": "EUR",
"fiat_rate": "1.10000000",
"mbtc": "20.34469",
"satoshi": 2034469,
"sats": "2034469",
"subaccount": 0,
"ubtc": "20344.69"
"is_current": true
}

```

fiat_currency Set to the users fiat currency if available, otherwise an empty string.

fiat_rate Set to the users fiat exchange rate if available, otherwise null.

is_current true if the "fiat_currency" and "fiat_rate" members are current.

2.55 Available currencies JSON

```

{
  "all": ["AUD", "BRL", "CAD", "CHF", "CNY", "DKK", "EUR", "GBP", "HKD", "IDR", "INR", "JPY", "MXN",
  ↪ "MYR", "NGN", "NOK", "NZD", "PLN", "RUB", "SEK", "SGD", "THB", "TRY", "USD", "ZAR"],
  "per_exchange": { "BITFINEX": ["USD"], "BITSTAMP": ["USD"], "BTC AVG": [], "BTC CHINA": [],
  ↪ "HUOBI": [], "KIWICOIN": ["NZD"], "KRAKEN": ["EUR", "USD"], "LOCALBTC": ["AUD", "BRL", "CAD",
  ↪ "CHF", "CNY", "DKK", "EUR", "GBP", "HKD", "IDR", "INR", "JPY", "MXN", "MYR", "NGN", "NOK", "NZD",
  ↪ "PLN", "RUB", "SEK", "SGD", "THB", "TRY", "USD", "ZAR"], "LUNO": ["IDR", "MYR", "NGN", "ZAR"],
  ↪ "QUADRIGACX": ["CAD", "USD"], "TRT": ["EUR"]}
}

```

2.56 HTTP parameters JSON

```

{
  "accept": "json"
  "method": "GET"
  "urls": [
    "https://assets.blockstream.info/index.json"
    "http://vi5flmr4z3h3luup.onion/index.json"
  ]
  "proxy": "localhost:9150"
  "headers": {"If-Modified-Since": "Mon, 02 Sep 2019 22:39:39 GMT"}
  "timeout": 10
}

```

2.57 Locktime details JSON

```

{
  "value": 65535
}

```

2.58 Asset parameters JSON

```
{
  "assets": true,
  "icons": true
}
```

2.59 Get assets parameters JSON

Information about Liquid assets can be obtained by either passing a list of asset ids to query:

```
{
  "assets_id": [ "6f0279e9ed041c3d710a9f57d0c02928416460c4b722ae3457a11eec381c526d",
    ↪ "ce091c998b83c78bb71a632313ba3760f1763d9cfcffae02258ffa9865a37bd2" ]
}
```

or by specifying one or more of the following attributes:

names a list of strings representing asset names;

tickers a list of strings representing asset tickers;

category must be one of: - "with_icons": only assets that have icons associated to them will be returned; - "hard_coded": only assets bundled in the GDK release will be returned; - "all": all the locally-stored assets and icons will be returned.

Specifying multiple attributes is interpreted as a logical AND. For example, {"category": "with_icons", "tickers": ["LCAD"]} will return all the assets with ticker LCAD that also have an icon.

2.60 Asset details JSON

```
{
  "assets": {
    "6f0279e9ed041c3d710a9f57d0c02928416460c4b722ae3457a11eec381c526d": {
      "asset_id": "6f0279e9ed041c3d710a9f57d0c02928416460c4b722ae3457a11eec381c526d"
      ↪ ,
      "contract": null,
      "entity": null,
      "issuance_prevout": {
        "txid": "0000000000000000000000000000000000000000000000000000000000000000"
      ↪ ,
        "vout": 0
      },
      "issuance_txin": {
        "txid": "0000000000000000000000000000000000000000000000000000000000000000"
      ↪ ,
        "vin": 0
      },
      "issuer_pubkey": "",
      "name": "btc",
      "precision": 8,
      "ticker": "L-BTC",
      "version": 0
    }
  }
}
```

(continues on next page)

(continued from previous page)

```

    }
  },
  "icons": {
    "6f0279e9ed041c3d710a9f57d0c02928416460c4b722ae3457a11eec381c526d": "BASE64"
  }
}

```

2.61 Error details JSON

```

{
  "details": "assertion failure: ga_session.cpp:2166:Unknown subaccount"
}

```

2.62 Get Subaccounts parameters JSON

Parameters controlling the *GA_get_subaccounts* call.

```

{
  "refresh": false
}

```

refresh If set to `true`, subaccounts are re-discovered if appropriate for the session type. Note that this will take significantly more time if set. Defaults to `false`.

2.63 Validate JSON

Passed to *GA_validate* to check the validity of gdk input data.

To validate addressees, for example prior to calling *GA_create_transaction*:

```

{
  "addressees": {},
  "network": "bitcoin"
}

```

addressees Mandatory. An array of *Addressee JSON* elements.

network Optional. The name of a network to validate the addressees against.

Validation includes that the address is correct and supported by the network, and that the amount given is valid. The given amount in whatever denomination will be converted into "satoshi" in the returned addressee. For Liquid, a valid hex "asset_id" must be present.

It is also possible to validate an addressee for another network than that of the current session. To do so, pass a network name in "network". Note that when validating against a different network, any amount in the addressee will not be validated or converted, as the session does not have pricing data for other networks than its own.

To validate a LiquiDEX version 1 proposal:

```
{
  "liquidex_v1": {
    "proposal": {}
  }
}
```

liquidex_v1/proposal The LiquiDEX version 1 proposal to validate.

2.64 Validate Result JSON

Returned from *GA_validate* to indicate the validity of the given JSON document.

```
{
  "is_valid": true,
  "errors": [],
  "addressees": {}
}
```

is_valid `true` if the JSON is valid, `false` otherwise.

errors An array of strings describing each error found in the given document; Empty if "is_valid" is `true`.

addressees If validating addressees, the given *Addressee JSON* elements with data sanitized and converted if required. For example, BIP21 URLs are converted to addresses, plus amount/asset if applicable.

GDK Create Transaction JSON

This section details how to create various kinds of transaction using *GA_create_transaction*. Once created, the resulting JSON is generally passed to *GA_blind_transaction* (on Liquid) to blind its outputs. Following this it is passed to *GA_sign_transaction* to obtain signatures, then broadcast to the network via *GA_send_transaction* or *GA_broadcast_transaction*.

3.1 Overview

The caller passes details about the transaction they would like to construct. The returned JSON contains the resulting transaction and an "error" element which is either empty if the call succeeded in creating a valid transaction or contains an error code describing the problem.

Building transactions can be done iteratively, by passing the result of one call into the next after making changes to the returned JSON. This is useful for interactive transaction creation as it allows users to see the effect of changes such as different fee rates interactively, and to fix errors on the fly. Because gdk will create new change addresses when constructing transactions, passing the result of the last call to *GA_create_transaction* makes each call significantly faster and more efficient compared to passing in the initial JSON.

When using gdk as a integration solution, *GA_create_transaction* is generally only called once, and if an error occurs the operation is aborted.

Note that the returned JSON will contain additional elements beyond those documented here. The caller should not attempt to change these elements; the documented inputs are the only user-level changes that should be made, and the internal elements may change name or meaning from release to release.

3.2 Mandatory and Optional Elements

Only two elements are always mandatory: "addressees" and "utxos". A transaction sending some amount from the wallet can be created using e.g:

```
{  
  "addressees": [ {} ],  
  "utxos": { }  
}
```

addressees Mandatory. An array of *Addressee JSON* elements, one for each recipient.

utxos Mandatory. The UTXOs to fund the transaction with, *Unspent outputs JSON* as returned by *GA_get_unspent_outputs*. Any UTXOs present are candidates for inclusion in the transaction.

Optional elements allow more precise control over the transaction:

fee_rate Defaults to the sessions default fee rate setting. The fee rate in satoshi per 1000 bytes to use for fee calculation.

utxo_strategy Defaults to "default". Set to "manual" for manual UTXO selection.

randomize_inputs Defaults to `true`. If set to `true`, the order of the used UTXOs in the created transaction is randomized.

is_partial Defaults to `false`. Used for creating partial/incomplete transactions such as half-swaps. If set to `true`, no change outputs will be created, fees will not be calculated or deducted from inputs, and the transaction inputs and outputs will not be expected to balance. Sets "randomize_inputs" to `false`. Consider using *GA_create_swap_transaction* instead of this element.

transaction_version Defaults to 2. The Bitcoin/Liquid transaction version to use.

transaction_locktime Defaults to The current block with occasional random variance for privacy. The transaction level locktime to use.

If you wish to customize a transaction further, consider creating a PSBT/PSET directly from the wallets inputs and using *GA_psbt_sign* to sign it.

3.3 Returned metadata

Some data returned when creating a transaction may be read by the user:

error If not empty, the error description that prevented the transaction being created.

transaction The hex-encoded resulting transaction. This may be partially complete or contain dummy data, e.g. missing blinding data or signatures before it is fully completed and signed. The `vsize` and `weight` elements described below are adjusted with reasonable estimates for any missing data until the transaction is fully signed.

transaction_vsize The expected final `vsize` of the "transaction" in vbytes.

transaction_weight The expected final weight of "transaction" in segwit weight units.

calculated_fee_rate The expected fee rate for the final signed transaction. This may differ slightly from the requested "fee_rate" due to variance in the size of witness data such as signatures.

transaction_inputs An array of the "utxos" elements that are used by the transaction.

transaction_outputs The transaction output details for the outputs of the transaction.

3.4 Addressee JSON

Describes an intended recipient for a transaction.

```
{
  "address": "2NFHMw7GbqnQ3kTYMrA7MnHiYDyLy4EQH6b",
  "satoshi": 100000,
  "asset_id": "6f0279e9ed041c3d710a9f57d0c02928416460c4b722ae3457a11eec381c526d",
  "is_greedy": false
}
```

address Mandatory. The address to send to. All address types for the network are supported. Additionally, [BIP 21](#) URLs are supported along with the [Liquid adaptation](#). Note that BIP 70 payment requests are not supported.

satoshi Normally mandatory. The amount to send to the recipient in satoshi. May be omitted when "is_greedy" is true.

asset_id Mandatory for Liquid, must not be present for Bitcoin. The asset to be sent to the recipient, in display hex format.

is_greedy Optional. If set to true, this addressee will consume any remaining value from UTXOs that match its "asset_id" (or all UTXOs in the case of Bitcoin). Only one greedy output per asset is allowed. When set to true, the value of the "satoshi" element is ignored and will be overwritten with the computed value.

Note: When an addressee represents a wallet address, the entire contents of the [GA_get_receive_address](#) call made to fetch the address must be passed. If only the address is provided, the addressee will be treated as an external addressee, and e.g. the summary values of the transaction in the top-level "satoshi" element will not be correct.

3.5 Coin selection

Callers can control the UTXOs used when creating a transaction. When using "utxo_strategy": "default", Bitcoin and Liquid-Bitcoin UTXOs are selected from the "utxos" element. The caller can reorder and filter these UTXOs using the query parameters to [GA_get_unspent_outputs](#) to control which UTXOs are used (and their ordering, if "randomize_inputs" is set to false). For asset UTXOs under Liquid, the default strategy attempts to select the minimum number of UTXOs to use without regard for their ordering in the "utxos" element.

For finer control, setting "utxo_strategy" to "manual" allows the UTXOs to be used to be placed directly into the "transaction_inputs" element by the caller. In this case, "utxos" is unused, and all given UTXOs will be added to the resulting transaction.

The sum of input UTXOs for a given asset must be sufficient to cover the amounts sent to any addressees receiving it, or an error will occur unless "is_partial" is true. Excess amounts will be either be returned to the wallet as change, or if an addressee has "is_greedy" set to true then the remainder will be assigned to it. When "utxo_strategy" is set to "default", UTXOs from the "utxos" element will be omitted from the created transaction if they are not needed.

Finally, creating a PSBT/PSET and using [GA_psbt_sign](#) to sign it allows exact specification of all transaction details including UTXOs.

3.6 Re-deposit

A re-deposit is just a standard sending transaction with the addressee being an address from the users wallet as returned from `GA_get_receive_address`, and `is_greedy` in the addressee being set to `"true"` to consume all of the UTXOs to redeposit.

3.7 Fee bump

A fee bump or RBF transaction increases the fee rate of an outgoing transaction that the caller has already submitted to the mempool, but which is not yet confirmed.

To create a fee bump, the caller should include the transaction to bump in the `"previous_transaction"` element. This must be in the format returned by `GA_get_transactions`. Additionally the new fee rate to bump to must be given in `"fee_rate"`, and any additional UTXOs to be used for providing the funds for the fee increase given in `"utxos"`.

```
{
  "previous_transaction": {},
  "fee_rate": 5000,
  "utxos": { }
}
```

previous_transaction The transaction to bump, as returned from *Transaction list JSON*.

fee_rate The new fee rate in satoshi per 1000 bytes to use for fee calculation. This must be higher than the exiting fee rate in `"previous_transaction"`.

3.8 Sweeping

A sweep transaction moves coins from an address with a known private key to another address. Unlike a simple send transaction, the coins to be moved are not associated with the users wallet in any way. Sweeping is typically used to move coins from a paper wallet into the users wallet.

To create a sweep transaction, use `GA_get_unspent_outputs_for_private_key` to fetch the unspent outputs to sweep. Including these in the `"utxos"` element when creating a transaction will automatically handle signing them.

Note that you should generally set `"is_greedy"` for one of the transaction addressees when sweeping to ensure that all of the UTXOs are spent together.

It is also possible to send the swept coin to an address that does not belong to the callers wallet. Currently it is not possible to include sweep inputs along with wallet inputs to combine spending.

This section describes the notifications emitted by the library.

All notifications contain an "event" element which describes the type of event being notified. The notification data is available under an element named with the content of the "event" element.

4.1 Network notification

Notified when the state of a session's underlying network connection changes.

```
{
  "event": "network",
  "network": {
    "wait_ms": 1000,
    "current_state": "disconnected",
    "next_state": "connected"
  }
}
```

current_state One of "connected" or "disconnected". The current state of the network connection.

next_state One of "connected" or "disconnected". The next state that the connection will move to. If this value is the same as "current_state" then no state change is currently in progress.

wait_ms The number of milliseconds before the current state will change to the next state. 0 if the change will happen immediately or no change is due to occur.

4.2 Tor notification

Notified when using the built-in Tor connection during connection establishment.

```
{
  "event": "tor",
  "tor": {
    "progress": 20,
    "summary": "Establishing an encrypted directory connection",
    "tag": "onehop_create"
  }
}
```

tor/progress An integer from 0-100 indicating the progress percentage.

tor/summary A human-readable summary of the current progress step.

tor/onehop_create A fixed identifier string for the current progress step.

4.3 Settings notification

Notified upon successful authentication. Describes the current wallet settings.

```
{
  "event": "settings",
  "settings": {}
}
```

settings Contains the *Settings JSON* of the user.

4.4 Two factor reset notification

Notified by multisig sessions upon successful authentication. Describes the current two factor reset status of the wallet.

```
{
  "event": "twofactor_reset",
  "twofactor_reset": {}
}
```

twofactor_reset Contains the "twofactor_reset" portion of *Two-Factor config JSON*.

4.5 Block notification

Notified when a new block is mined by the network.

```
{
  "event": "block",
  "block": {
    "block_hash": "00000000a09b62cc7c076cf8bb25840e67bb5f9f47492f8a82a09105a6aab72d",
    "block_height": 2138311,
    "initial_timestamp": 1489943482,
    "previous_hash":
    ↪ "00000000000000bcbf344da3c3d691f5581136bf78c52de4c712949541f0ccf3c"
  }
}
```

block/block_hash The hash of the block.

block/block_height The height of the block.

block/initial_timestamp Multisig only. The time that the users wallet was created, in seconds since the epoc.

block/previous_hash The hash of the block prior to this block.

4.6 Transaction notification

Notified when a new transaction is received by the wallet.

```
{
  "event": "transaction",
  "transaction": {
    "satoshi": 50000,
    "subaccounts": [
      0
    ],
    "txhash": "2bee55e07ab6cc520487f57cb74e87c2960d5f01d291d34f6b395417a276a42c",
    "type": "incoming"
  }
}
```

transaction/satoshi Bitcoin only. The net amount of the transaction (always positive).

transaction/subaccounts The wallet subaccounts the transaction affects.

transaction/txhash The txid of the transaction.

transaction/type Bitcoin only. One of "incoming", "outgoing" or "redeposit".

4.7 Ticker notification

Notified when the user's exchange rate changes.

```
{
  "event": "ticker",
  "ticker": {
    "currency": "NZD",
    "exchange": "KIWICOIN",
    "rate": "44100.84"
  }
}
```

ticker/currency The user's chosen fiat currency.

ticker/exchange The user's chosen exchange source.

ticker/rate The price of 1 Bitcoin expressed in the user's fiat currency, expressed as a floating point string.

GDK Hardware Wallet Interface

This section details the format of data requests from hardware wallet interaction during resolution of `GA_auth_handler` processing when `GA_auth_handler_get_status` returns the status `"resolve_code"` with a `"required_data"` element present.

5.1 Required Data JSON

Returned as an element `"required_data"` of *Auth handler status JSON* when data is required from a registered hardware device.

```
{
  "action": "",
  "device": {
  },
}
```

action Describes the hardware wallet data requested.

device Contains the *HW device JSON* originally registered with the session.

Additional fields will be present depending on the action requested, as follows:

5.2 Hardware Get XPubs Action

When `"action"` is `"get_xpubs"`, this describes a request to compute one or more xpubs from the wallet's master key.

```
{
  "paths": [ [], [ 2147501889 ] ]
}
```

paths An array of unsigned 32-bit integer arrays representing each xpub to fetch. The integer values should be interpreted per BIP32, i.e. the topmost bit may be set to indicate a private derivation in the path. An empty array indicates that the top level xpub should be returned.

Expected response:

```
{
  "xpubs": [
    ↪ "tpubD8G8MPH9RK9uk4EV97RxhzaY8SJPuWxNViHUWji92i8B7vYdht797PPDrJveethnKxonJe8SbaScAC1YJ8xAzZbH9Uvy
    ↪ ",
    ↪ "tpubD6NzVbkrYhZ4X9jwmpJxg1kjEJTQgkrnHNEWww2e86X1eUfWu1f7hZpgezAyWUk5zRt4fMPHB33CXrvJSYHHAoVMFXrfx
    ↪ "
  ]
}
```

xpubs An array of base58-encoded BIP32 extended public keys, in the same order as the "paths" elements in the request.

5.3 Hardware Get Master Blinding Key Action

When "action" is "get_master_blinding_key", this describes a request to return the wallet's SLIP0077 master blinding key if the user allows this.

Note: This action is only returned when using the Liquid network.

No request data is currently associated with this request.

Expected response:

```
{
  "master_blinding_key":
  ↪ "512cd6c0b73452a2414e9d86d37cdcc8283b44f0b6dd2b1eec23c59ff12b4f7e5949569b3430220dafce1e0e299a2a6f3
  ↪ "
}
```

master_blinding_key The 512-bit or 256-bit master blinding key for the wallet, hex-encoded. If a 256-bit key is returned, it should be the lower 256-bits of the SLIP0021 derived node as specified in <https://github.com/satoshilabs/slips/blob/master/slip-0077.md>.

Note: If the user denies the request to share the key, an empty string should be returned.

5.4 Hardware Sign Message Action

When "action" is "sign_message", this describes a request to sign a message using the given path.

```
{
  "message": "A text message to sign",
  "path": [ 1195487518 ],
```

(continues on next page)

(continued from previous page)

```

"use_ae_protocol": false,
"create_recoverable_sig": false
}

```

message The message to be utf-8 encoded and signed.

path The path from the wallet's master key to the key that the message should be signed with.

use_ae_protocol `true` if the hardware device advertises Anti-Exfil support and it should be used for signing, `false` otherwise.

create_recoverable_sig `true` if the signature to produce should be recoverable. Default `false`.

Expected response:

```

{
  "signature":
  ↪ "304402207c673ef4255873cf095016c98c4982cea9a5133060b66a380f1bf3880e54f6c8022056fd731cbdd44cd9636621
  ↪ "
}

```

signature The ECDSA signature corresponding to the given message. If "create_recoverable_sig" is `false` it must use DER encoding, otherwise it must be encoded in hex.

5.5 Hardware Get Blinding Public Keys Action

When "action" is "get_blinding_public_keys", this describes a request to compute blinding public keys from wallet scripts.

Note: This action is only returned when using the Liquid network.

```

{
  "scripts": [ "a91403f650e2434916d5b7f124de8f673442b696282887" ]
}

```

scripts An array of hex-encoded scripts for which a blinding key should be generated.

Expected response:

```

{
  "public_keys": [ "02045e92b8f68bd066180c05a39969f862a67f4efc8f5d7aeca32c627a463b8f27
  ↪ " ]
}

```

public_keys An array of hex-encoded compressed public keys for blinding the given scripts.

5.6 Hardware Get Blinding Nonces Action

When "action" is "get_blinding_nonces", this describes a request to compute blinding nonces and possibly blinding public keys for the given scripts and shared public keys.

Note: This action is only returned when using the Liquid network.

```
{
  "blinding_keys_required": true
  "scripts": [ "a91403f650e2434916d5b7f124de8f673442b696282887" ],
  "public_keys": [ "035f242d49b88ca17948b156263e1f0c86d2cc9e9ff316b058dbbdb351e34bc9aa
  ↪ " ]
}
```

blinding_keys_required true if the blinding public keys must be returned, false otherwise. Blinding public keys are not requested if the master blinding key has previously been given.

public_keys An array of hex-encoded compressed shared public keys for computing the nonces.

scripts An array of hex-encoded scripts for which a blinding key should be generated and then the nonce computed using the public key given.

Expected response:

```
{
  "public_keys": [ "02045e92b8f68bd066180c05a39969f862a67f4efc8f5d7aeca32c627a463b8f27
  ↪ " ]
  "nonces": [ "8d940a5ec4ad122394cd2596ecfbf933a8d8fb0196015cc0a35399e3c326758c" ]
}
```

public_keys An array of hex-encoded compressed public keys for blinding the given scripts. Must be present if "blinding_keys_required" was true in the request, and absent otherwise.

nonces An array of hex-encoded 256 bit blinding nonces.

5.7 Hardware Get Blinding Factors Action

When "action" is "get_blinding_factors", this describes a request to compute asset (ABF) and value (VBF) blinding factors for the given transaction outputs.

Note: This action is only returned when using the Liquid network.

```
{
  "is_partial": false,
  "transaction_outputs": [],
  "transaction_inputs": [
    {
      "txhash": "797c40d53c4a5372303f765281bb107c40ed9618646c46851514ff0483bee894"
      "pt_idx": 2,
    },
    {
      "txhash": "9c7cffca5711968a22b8a03cc6d17224d0d85d884a4d2f638371b6fd6d59afdb"
      "pt_idx": 1,
    }
  ]
}
```

is_partial true if transaction is incomplete, e.g. one half of a swap transaction.

transaction_outputs The transaction output details for the outputs to be blinded, in the format returned by *GA_create_transaction*. Any output with a "blinding_key" key present requires blinding factors to be returned. When "is_partial" is false, the final vbf need not be returned. An empty string should be returned for blinding factors that are not required. It is not an error to provide blinding factors that are not required; they will be ignored.

transaction_inputs An array of prevout txids and their indices, supplied so the request handler can compute hashPrevouts for deterministic blinding.

Expected response:

```
{
  "amountblinders": [
    "ce8259bd2e7fa7d6695ade7cf8481919612df28e164a9f89cd96aace69a78bb9",
    ""
  ],
  "assetblinders": [
    "5ca806862967cde0d51950dd4e9add68e7cae8cda928750037fca1fb9cfc9e58",
    "5748810a8d2c4d87ea8c3038fb71369d8d9c85f09cfa4f6412359910fce93616"
  ]
}
```

amountblinders An array of hex-encoded, display format value blinding factors (VBFs) to blind the transaction output values. Any non-required values should be returned as empty strings.

assetblinders An array of hex-encoded, display format asset blinding factors (ABFs) to blind the transaction output assets. Any non-required values should be returned as empty strings.

5.8 Hardware Sign Transaction Action

When "action" is "sign_tx", this describes a request to sign one or more inputs of a transaction.

```
{
  "transaction":
  ↪ "02000000000010135d2bb82963e54a9060567b101760530797590d2b4a636606c4f1e6ac62bed4300000000230000000000
  ↪ "
  "transaction_inputs": [],
  "transaction_outputs": [],
  "use_ae_protocol": false,
  "is_partial": false
}
```

transaction The hex-encoded transaction to sign.

transaction_inputs Contains details of each of the inputs in the transaction, in the format returned by *GA_create_transaction*. Any input without a "skip_signing" key present and set to true requires a signature to be returned. An empty string should be returned for signatures that are not required.

transaction_outputs The transaction output details for the outputs to be signed, in the format returned by *GA_create_transaction*.

use_ae_protocol true if the hardware device advertises Anti-Exfil support and it should be used for signing, false otherwise.

is_partial true if transaction is incomplete, e.g. one half of a swap transaction.

Expected response:

```
{
  "signatures": [
    ↪ "30440220580c7ef934d5d8f31c1c592fbf0e5bc3267b76995206f0eb61616eb2f8f6e1c4022022e3feaf88469328bdaff3
    ↪ " ]
}
```

signatures The ECDSA signatures corresponding to each input in the request, hex-encoded from the DER representation plus sighash byte.

CHAPTER 6

GDK Swap Interface

This section details the Liquid swap protocols supported by GDK and the functions available to run those protocols. Currently there is only one swap protocol supported, *LiquiDEX*.

Warning: Note that in the current version (1) of the protocol if the swap involves inputs or outputs from a "2of2_no_recovery" account is not safe to send the proposal directly to untrusted parties (either directly to the Taker, or an untrusted third party). Unless you know what you are doing you should not use this version of the protocol.

LiquiDEX is a 2-step swap protocol, to perform a swap of this kind use "swap_type" "liquidex".

The protocol is started by the Maker, who creates a proposal to swap a certain utxo for a certain amount of another asset. This action is performed using *GA_create_swap_transaction*.

The proposal is then shared with the Taker, who first validates it and verifies that it is a swap that it is willing to accept. This action is performed using *GA_validate*.

If the Taker wants to accept the proposal, they will add more inputs and outputs to fund and balance the transaction. This action is performed using *GA_complete_swap_transaction*.

Note that, unlike *GA_create_swap_transaction*, *GA_complete_swap_transaction* requires the caller to sign the transaction with *GA_sign_transaction*. Moreover for "2of2_no_recovery" (AMP) subaccounts, the caller should get a delayed signature for the Maker input.

7.1 LiquiDEX Create Swap transaction JSON

"input_type" and "output_type" must be "liquidex_v1".

```
{
  "swap_type": "liquidex",
  "input_type": "liquidex_v1",
  "output_type": "liquidex_v1",
  "liquidex_v1": {
```

(continues on next page)

(continued from previous page)

```
"receive": [{
  "asset_id": "ASSET_ID",
  "satoshi": 1
}],
"send": [{}],
},
}
```

receive/asset_id The hex-encoded asset id to receive, in display format. This list must have 1 element.

receive/satoshi The satoshi amount of the specified asset to receive. This list must have 1 element.

send The Maker's UTXO to swap, as returned from *GA_get_unspent_outputs*. This list must have 1 element. The swapped asset will be received to the same subaccount as the utxo provided.

7.2 LiquiDEX Create Swap Transaction Result JSON

Returned when "output_type" is "liquidx_v1".

```
{
  "liquidx_v1": {
    "proposal": {},
  }
}
```

proposal The LiquiDEX version 1 proposal to be shared.

7.3 LiquiDEX Complete Swap transaction JSON

"input_type" must be "liquidx_v1", and "output_type" must be "transaction".

```
{
  "swap_type": "liquidx",
  "input_type": "liquidx_v1",
  "output_type": "transaction",
  "utxos": {},
  "liquidx_v1": {
    "proposals": [{}],
  },
}
```

proposals The LiquiDEX version 1 proposals to take.

CHAPTER 8

Indices and tables

- `genindex`
- `search`

G

- GA_ack_system_message (*C function*), 17
- GA_auth_handler_call (*C function*), 19
- GA_auth_handler_get_status (*C function*), 18
- GA_auth_handler_request_code (*C function*), 19
- GA_auth_handler_resolve_code (*C function*), 19
- GA_bcur_decode (*C function*), 21
- GA_bcur_encode (*C function*), 21
- GA_blind_transaction (*C function*), 12
- GA_broadcast_transaction (*C function*), 14
- GA_change_settings (*C function*), 17
- GA_change_settings_twofactor (*C function*), 19
- GA_complete_swap_transaction (*C function*), 13
- GA_connect (*C function*), 2
- GA_convert_amount (*C function*), 10
- GA_create_session (*C function*), 1
- GA_create_subaccount (*C function*), 5
- GA_create_swap_transaction (*C function*), 12
- GA_create_transaction (*C function*), 11
- GA_decrypt_with_pin (*C function*), 11
- GA_destroy_auth_handler (*C function*), 19
- GA_destroy_json (*C function*), 18
- GA_destroy_session (*C function*), 2
- GA_destroy_string (*C function*), 22
- GA_disable_all_pin_logins (*C function*), 11
- GA_encrypt_with_pin (*C function*), 10
- GA_generate_mnemonic (*C function*), 22
- GA_generate_mnemonic_12 (*C function*), 22
- GA_get_assets (*C function*), 3
- GA_get_available_currencies (*C function*), 10
- GA_get_balance (*C function*), 10
- GA_get_credentials (*C function*), 16
- GA_get_fee_estimates (*C function*), 16
- GA_get_networks (*C function*), 23
- GA_get_previous_addresses (*C function*), 8
- GA_get_proxy_settings (*C function*), 2
- GA_get_random_bytes (*C function*), 22
- GA_get_receive_address (*C function*), 8
- GA_get_settings (*C function*), 17
- GA_get_subaccount (*C function*), 6
- GA_get_subaccounts (*C function*), 6
- GA_get_system_message (*C function*), 16
- GA_get_thread_error_details (*C function*), 1
- GA_get_transaction_details (*C function*), 9
- GA_get_transactions (*C function*), 7
- GA_get_twofactor_config (*C function*), 17
- GA_get_uniform_uint32_t (*C function*), 23
- GA_get_unspent_outputs (*C function*), 8
- GA_get_unspent_outputs_for_private_key (*C function*), 9
- GA_get_wallet_identifier (*C function*), 3
- GA_get_watch_only_username (*C function*), 5
- GA_http_request (*C function*), 3
- GA_init (*C function*), 1
- GA_login_user (*C function*), 4
- GA_psbt_get_details (*C function*), 13
- GA_psbt_sign (*C function*), 13
- GA_reconnect_hint (*C function*), 2
- GA_refresh_assets (*C function*), 3
- GA_register_network (*C function*), 23
- GA_register_user (*C function*), 4
- GA_remove_account (*C function*), 5
- GA_rename_subaccount (*C function*), 6
- GA_send_nlocktimes (*C function*), 15
- GA_send_transaction (*C function*), 14
- GA_set_csvtime (*C function*), 15
- GA_set_nlocktime (*C function*), 15
- GA_set_notification_handler (*C function*), 1
- GA_set_transaction_memo (*C function*), 15
- GA_set_unspent_outputs_status (*C function*), 9
- GA_set_watch_only (*C function*), 5
- GA_sign_message (*C function*), 14
- GA_sign_transaction (*C function*), 12
- GA_twofactor_cancel_reset (*C function*), 20

`GA_twofactor_change_limits` (*C function*), [21](#)
`GA_twofactor_reset` (*C function*), [20](#)
`GA_twofactor_undo_reset` (*C function*), [20](#)
`GA_update_subaccount` (*C function*), [7](#)
`GA_validate` (*C function*), [4](#)
`GA_validate_asset_domain_name` (*C function*),
[4](#)
`GA_validate_mnemonic` (*C function*), [22](#)