LNP/BP Standards Association

RGB v0.10 release, part 3

Dr Maxim Orlovsky
Chief engineering officer at **LNP/BP Standards Association**,

@dr_orlovsky

FBDE A843 3DDC 1E69 FA90 C35E FFC0 2509 47E5 C6F7

RGB v0.10 release progress

	Readiness	Released	Presentation
Consensus (Core lib)	✅	9 Feb	Part 1
Standard library	✅	9-10 March	Part 2
Wallet library	✅	22 Match	Part 3
Command-line tool	🟠	ETA 30 March	Part 4

What’s new (wallet library)

- Working with PSBT
- Creating & accepting transfer consignments
  (now with WASM support and without database connectivity!)
- Invoicing

What’s new (wallet library)

- secp256k1 0.28
- rust-bitcoin 0.30

New invoices

URL scheme

rgb:DirectDetectEqual0EKkb7TMfbPxzn7UhvXqhoCutzdZkSZCNYxVAVjsA67fW/
RGB20/100@6kzbKKffP6xftkxn9UP8gWqiC41W16wYKE5CYaVhmEve

Invoices are URLs!

That should make life much easier

URL schemes

- **rgb**: unspecified transport (offline, messaging, e-mails etc)
- **rgb-rpc**:*//server.name/* – JSON RGB RPC protocol
- **rgb+http(s)**:*//server.name/* – REST protocol
- **rgb+ws(s)**:*//server.name/* – WebSockets protocol
- **rgb+storm**:*//node_key@node_addr/* – LN-based Storm protocol

New invoices

rgb:DirectDetectEqual0EKkb7TMfbPxzn7UhvXqhoCutzdZkSZCNYxVAVjsA67fW/
RGB20/100@6kzbKKffP6xftkxn9UP8gWqiC41W16wYKE5CYaVhmEve

rgb+https://mycitadel.org/
DirectDetectEqual0EKkb7TMfbPxzn7UhvXqhoCutzdZkSZCNYxVAVjsA67fW/
RGB20/100@6kzbKKffP6xftkxn9UP8gWqiC41W16wYKE5CYaVhmEve
?sig=6kzbKKffP6xftkxn9UP8gWqiC41W16wYKE5CYaVhmEve#random-bloody-morning

New invoices

Contract info (asset):

Contract mnemonic

Visual separator

Contract identifier in Base58

rgb:DirectDetectEqual0EKkb7TMfbPxzn7UhvXqhoCutzdZkSZCNYxVAVjsA67fW/
RGB20/100@6kzbKKffP6xftkxn9UP8gWqiC41W16wYKE5CYaVhmEve

New invoices

```
rgb:DirectDetectEqual0EKkb7TMfbPxzn7UhvXqhoCutzdZkSZCNYxVAVjsA67fW/
RGB20/100@6kzbKKffP6xftkxn9UP8gWqiC41W16wYKE5CYaVhmEve
```

Interface name

New invoices

rgb:DirectDetectEqual0EKkb7TMfbPxzn7UhvXqhoCutzdZkSZCNYxVAVjsA67fW/
RGB20/100@6kzbKKffP6xftkxn9UP8gWqiC41W16wYKE5CYaVhmEve

Transferred state (amount)

Blinded UTXO

Assignment

New invoices

URL scheme

Contract mnemonic

Visual separator

Contract identifier in Base58

rgb:DirectDetectEqual0EKkb7TMfbPxzn7UhvXqhoCutzdZkSZCNYxVAVjsA67fW/

RGB20/100@6kzbKKffP6xftkxn9UP8gWqiC41W16wYKE5CYaVhmEve

Interface name

Transferred state (amount)

Blinded UTXO

New invoices

- Invoices are URLs: can be opened with one click in the wallet app
- All important information is readable and visually distinguishable
- Simplified checks with mnemonic words
- Shortened Base58 encoding (comparing to Bech32)

Why not Bech32(m)?

- Too lengthy without advantages:
- Not better for QR codes
- Do not work as URLs
- Hard to check (most people do not know how)
- Can’t be longer than 90 chars
  (LN invoices is a technical nosence)
- Pointless error-correction code instead of checksum

URL ubiquity

This pays invoice:

rgb:DirectDetectEqual0EKkb7TMfbPxzn7UhvXqhoCutzdZkSZCNYxVAVjsA67fW/RGB20/100@6kzbKKffP6xftkxn9UP8gWqiC41W16wYKE5CYaVhmEve

URL ubiquity

This opens contract in a default RGB app/wallet **using RGB20 interface:**

rgb:DirectDetectEqual0EKkb7TMfbPxzn7UhvXqhoCutzdZkSZCNYxVAVjsA67fW/RGB20

This pays invoice:

rgb:DirectDetectEqual0EKkb7TMfbPxzn7UhvXqhoCutzdZkSZCNYxVAVjsA67fW/RGB20/100@6kzbKKffP6xftkxn9UP8gWqiC41W16wYKE5CYaVhmEve

URL ubiquity

This opens contract in a default RGB app/wallet:

rgb:DirectDetectEqual0EKkb7TMfbPxzn7UhvXqhoCutzdZkSZCNYxVAVjsA67fW

This opens contract in a default RGB app/wallet using RGB20 interface:

rgb:DirectDetectEqual0EKkb7TMfbPxzn7UhvXqhoCutzdZkSZCNYxVAVjsA67fW/RGB20

This pays invoice:

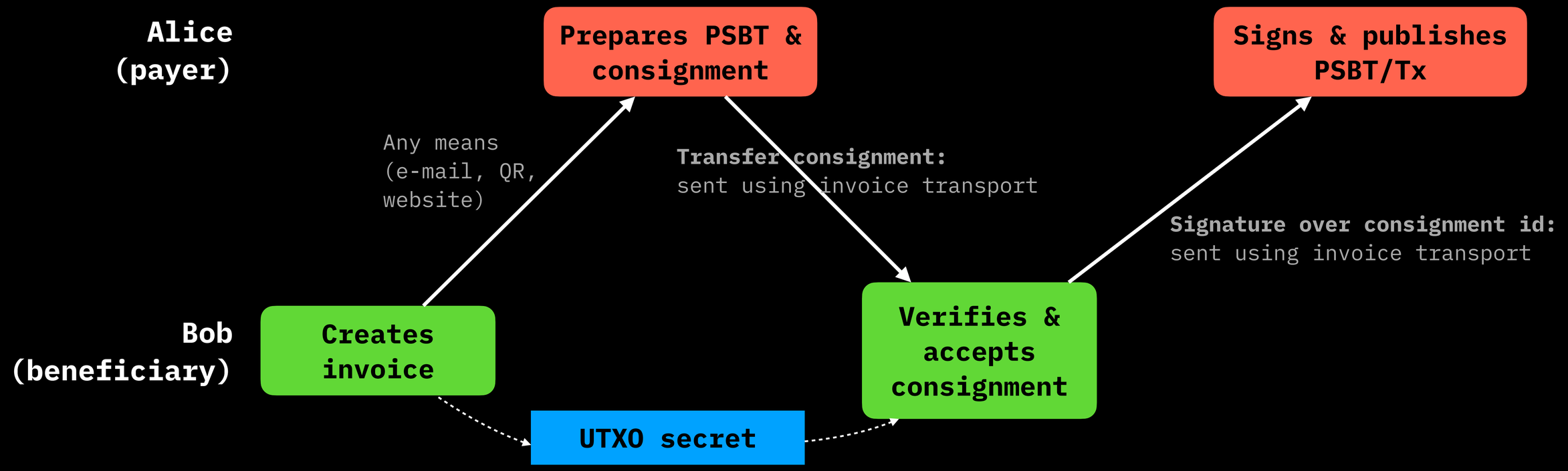
rgb:DirectDetectEqual0EKkb7TMfbPxzn7UhvXqhoCutzdZkSZCNYxVAVjsA67fW/RGB20/100@6kzbKKffP6xftkxn9UP8gWqiC41W16wYKE5CYaVhmEve

Invoices can handle much more!

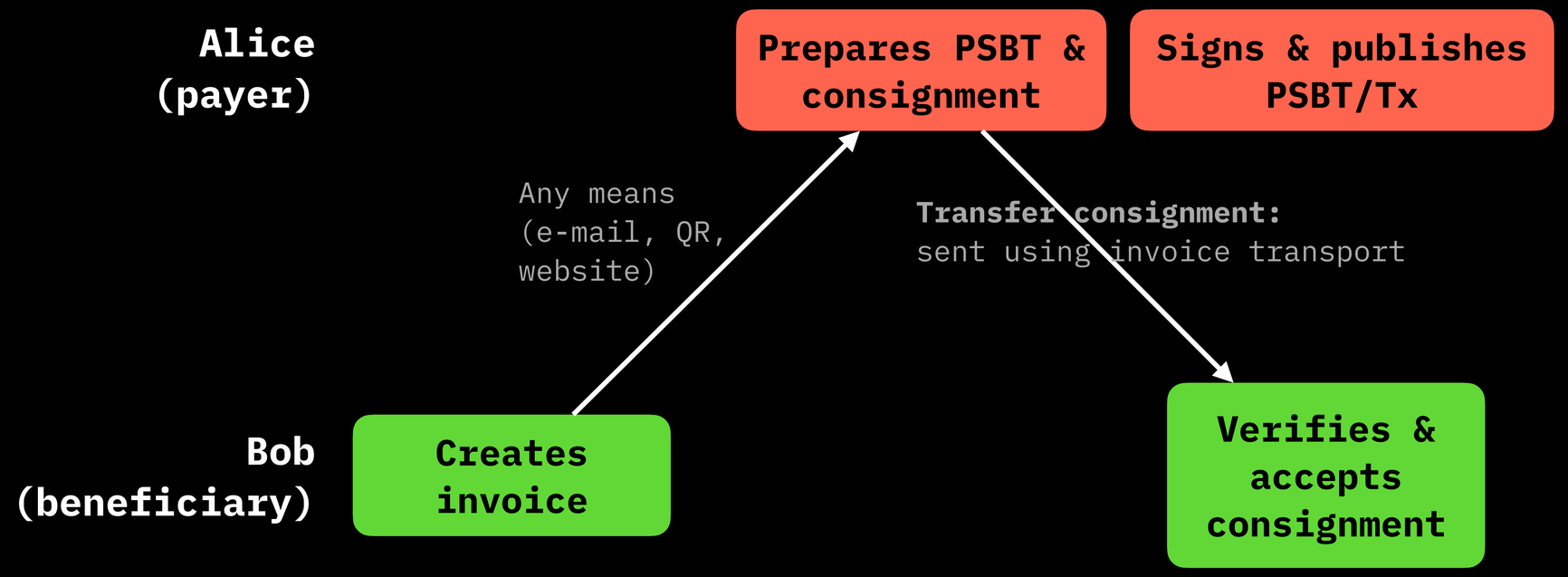
NFTs: rgb:DirectDetectEqual0EKkb7TMfbPxzn7UhvXqhoCutzdZkSZCNYxVAVjsA67fW/RGB21/DbwzvSu4BZU81jEpE9FVZ3xjcyuTKWWy2gmdnaxtACrS@6kzbKKffP6xftkxn9UP8gWqiC41W16wYKE5CYaVhmEve

Issuing: rgb:DirectDetectEqual0EKkb7TMfbPxzn7UhvXqhoCutzdZkSZCNYxVAVjsA67fW/RGB20/issue/1000000@6kzbKKffP6xftkxn9UP8gWqiC41W16wYKE5CYaVhmEve

Transfer workflow



Transfer workflow



Transfer workflow

```
alice$ rgb invoice RGB20 100 USDT tapret1st:456e3..dfe1:0
```

Transfer workflow

```
alice$ rgb invoice RGB20 100 USDT tapret1st:456e3..dfe1:0
```

```
rgb:DirectDetectEqual0EKkb7TMfbPxzn7UhvXqhoCutzdZkSZCNYxVAVjsA67fW/
RGB20/100@6kzbKKffP6xftkxn9UP8gWqiC41W16wYKE5CYaVhmEve
```

Transfer workflow

```
alice$ rgb invoice RGB20 100 USDT tapret1st:456e3..dfe1:0

       rgb-rpc:DirectDetectEqual0EKkb7TMfbPxzn7UhvXqhoCutzdZkSZCNYxVAVjsA67fW/
       RGB20/100@6kzbKKffP6xftkxn9UP8gWqiC41W16wYKE5CYaVhmEve

  bob$ wallet construct tx.psbt
  bob$ rgb transfer tx.psbt <invoice> consignment.rgb
```

Transfer workflow

```
alice$ rgb invoice RGB20 100 USDT tapret1st:456e3..dfe1:0

        rgb:DirectDetectEqual0EKkb7TMfbPxzn7UhvXqhoCutzdZkSZCNYxVAVjsA67fW/
        RGB20/100@6kzbKKffP6xftkxn9UP8gWqiC41W16wYKE5CYaVhmEve

  bob$ wallet construct tx.psbt
  bob$ rgb transfer tx.psbt <invoice> consignment.rgb

alice$ rgb accept consignment.rgb
```

Transfer workflow

```
alice$ rgb invoice RGB20 100 USDT tapret1st:456e3..dfe1:0

       rgb:DirectDetectEqual0EKkb7TMfbPxzn7UhvXqhoCutzdZkSZCNYxVAVjsA67fW/
       RGB20/100@6kzbKKffP6xftkxn9UP8gWqiC41W16wYKE5CYaVhmEve

  bob$ wallet construct tx.psbt
  bob$ rgb transfer tx.psbt <invoice> consignment.rgb

alice$ rgb accept consignment.rgb

       DbwzvSu4BZU81jEpE9FVZ3xjcyuTKWWy2gmdnaxtACrS # <— signature
```

Transfer workflow

```
alice$ rgb invoice RGB20 100 USDT tapret1st:456e3..dfe1:0

       rgb:DirectDetectEqual0EKkb7TMfbPxzn7UhvXqhoCutzdZkSZCNYxVAVjsA67fW/
       RGB20/100@6kzbKKffP6xftkxn9UP8gWqiC41W16wYKE5CYaVhmEve

  bob$ wallet construct tx.psbt
  bob$ rgb transfer tx.psbt <invoice> consignment.rgb

alice$ rgb accept consignment.rgb

       DbwzvSu4BZU81jEpE9FVZ3xjcyuTKWWy2gmdnaxtACrS # <- signature

  bob$ rgb check <sig>
  bob$ wallet sign -publish tx.psbt
```

Previous transfer script

```
# -- PAYMENT ------------------------------------------------
# Go back to the original machine

# Also save the beneficiary txo string
TXOB="txob1....."
UTXO_SRC=$UTXO_ISSUE # We will transfer issued funds, but in fact it can be any UTXO with the asset

# First, we compose consignment describing the asset we have on _our existing UTXO_
# (this is the UTXO we issued asset to, but it can be any mined UTXO having asset).
# This is not the final consignment; it is a base for constructing the consignment.
CONSIGNMENT=${DIR}/demo.rgbc
rgb-cli -n testnet transfer compose ${CONTRACT_ID} ${UTXO_SRC} ${CONSIGNMENT}
# We can verify that the consignment is correct
rgb consignment validate ${CONSIGNMENT}

# Next, we need to compose state transition performing the transfer for our contract.
# We do not need stash for that, since the base consignment we just created contains
# all required state information. We use RGB20 utility which understands concept of
# the fungible asset and can prepare state transition according to RGB20 schema rules.
TRANSITION=${DIR}/demo.rgbt
rgb20 transfer --utxo ${UTXO_SRC} --change 9900@tapret1st:${UTXO_CHANGE} \
      ${CONSIGNMENT} 100@${TXOB} ${TRANSITION}

# Now we need to prepare PSBT file containing the witness transaction, which will
# commit to the transfer we are doing. We also need to allow Tapret commitments in
# the first output (which is the change output created automatically).
FEE=500
PSBT="${DIR}/demo.psbt"
btc-cold construct --input "${UTXO_SRC} /0/0" --allow-tapret-path 1 ${WALLET} ${PSBT} ${FEE}

# Now we need to embed information about the contract into PSBT
rgb-cli -n testnet contract embed ${CONTRACT_ID} ${PSBT}

# We need to add to the PSBT information about the state transition.
# The daemon will also analyze are there any other assets (under different contracts)
# on the UTXOs we spend in PSBT, and if any, it will generate "blank" state transitions,
# which will be also added to the PSBT file together with contracts for each of those
# assets. Finally, the node will generate disclosure with all those other assets moved
# and store it internally to update its stash once the transaction from PSBT gets
# finalized and mined.
rgb-cli -n testnet transfer combine ${CONTRACT_ID} ${TRANSITION} ${PSBT} ${UTXO_SRC}

# This processes all state transitions under all contracts which are present in PSBT
# and prepares information about them which will be used in LNPBP4 commitments.
rgb psbt bundle ${PSBT}

# We can analyze PSBT and see all the details we added to it
rgb psbt analyze ${PSBT}
```

```
rgb-cli -n testnet transfer finalize --endseal ${TXOB} ${PSBT} ${CONSIGNMENT} --send $BENEFICIARY

# Those who interested can look into the transfer consignment
rgb consignment inspect ${CONSIGNMENT}

# If we validate the consignment now, we will see that it will report absence
# of the mined endpoint transaction, which is correct - we have not yet published
# witness transaction from the PSBT file
rgb consignment validate ${CONSIGNMENT}

# Lets finalize, sign & publish the witness transaction
btc-hot sign ${PSBT} ${DIR}/testnet
btc-cold finalize --publish testnet ${PSBT}

# Now, once the transaction will be mined, the verification should pass
rgb consignment validate ${CONSIGNMENT}

# -- CONSUME AND UNLOCK ASSET -------------------------------------------------
# Go to a remote server / other machine and do the following

# First, we must get all the information required to consume the consignment file.
# The consignment file
CONSIGNMENT=${DIR}/demo.rgbc
# The UXTO used in blinded utxo operation.
RECEIVE_UXTO=$UTXO
# The blinding factor generated in blinded utxo operation.
BLINDING_FACTOR=$INVOICE_BLINDING
# The close method used in PAYMENT operation.
CLOSE_METHOD="tapret1st"

# Next, we need to compose the reveal information to unlock the uxto.
REVEAL="$CLOSE_METHOD@$RECEIVE_UXTO#$BLINDING_FACTOR"

# Let's consume and reveal the concealed seal inside the consignment file.
rgb -n testnet transfer consume ${CONSIGNMENT} --reveal ${REVEAL}

# Now, we need to check if contract state has changed.
# First, get the contract ID
rgb -n testnet contract list
CONTRACT_ID="rgb1...."

# Next, check the new contract state
rgb -n testnet contract state ${CONTRACT_ID}
```

…and one more thing…