VRF Lottery

This application uses VRF to confirm the randomness of various online lotteries.

The unique provable random index created during the VRF scheme is used to select the winners. The list of participants is shuffled using the Fisher–Yates shuffle algorithm and N first winners are selected. Now the tweeter and waves asset recipients support are made, but you can add any source of participants of the lottery in the future. The applications use an array of strings as a list of participants.

Installation

Download the appropriate file for your platform from the Releases section (windows, mac os or linux).

Scheme of the lottery

To start the lottery, you need to create a curve25519 public and private keys (you can use `create-keys` tool from this package), select the height of the Waves block in the future (you can use `waves-height-tools` tool from this package), the signature of which will participate in the lottery. In addition to the conditions for participation in the draw, you need to publicly declare the public key and the height that participants will use in the future for verification. At the time of summing up the results, the lottery organizer creates a list of participants (you can use `outgoing-waves-tx-recipients` or `retweets-parser` tool from this package or create an array of participants manually), adds the block signature at a pre-selected height (saves it as a 'provable file'), selects the winners (using `pick-winner` tool from this package) and creates a 'proof' bytes strings). When announcing the results, the organizer publishes the 'provable file' (list of participants and the block signature) and 'proof'. Using the public key published at the beginning of the lottery, 'provable file' and 'proof', anyone can check the winners (using `verify-winner` tool from this package), the presence of himself in the list of participants and the impartiality of the organizer.

Since the lottery organizer cannot predict the block signature in the future, he cannot know in advance the pseudo-random value that will be involved in the lottery. Only the miner of the block in the future may in theory know his signature in advance, but he does not have the private key of the lottery organizer to manipulate it to win the contest, so he does not know how the signature affects the winners.

Description

The package contains convenient console apps to create keys, calculate waves height at time and a time from the waves height, load the participants lists (from twitter or waves token recipients), select the winners and check the winners.

Also there're 5 useful native applications:

App `create-keys`

```
$ ./create-keys -h
  Usage of ./create-keys:
    -json bool
        Output JSON, not plain text
```

App `waves-height-tools`

```
$ ./waves-height-tools -h
Usage of ./waves-height-tools:
  -at string
        Calculate height at datetime
  -block int
        Calculate datetime by height
```

App `pick-winner`

```
$ ./pick-winner -h
 Usage of ./pick-winner:
   -blockHeight uint
        Waves block height, the signature of it will be used in provable
message
   -participantsFile string
        Path to file with participants, it should contains json array of
strings
   -pickN uint
        Number of winners to pick, it should be >= 1 (default 1)
   -privateKey string
        curve25519 private key in Base58 to prove the message
   -json bool
        Output JSON, not plain text
```

App `verify-winner`

```
$ ./verify-winner -h
Usage of ./verify-winner:
  -blockHeight uint
        Waves block height, the signature of it will be used to validate
the data
  -pickN uint
        Number of winners to pick, it should be >= 1
  -proof string
        Proof bytes in Base58 to validate the message
  -provableFile string
        Path to file to validate, it should contains 2 lines: json array
of the participants, '\n' line separator and a block signature
  -publicKey string
        curve25519 public key in Base58 to validate the message
  -json bool
        Output JSON, not plain text
```

App `retweets-parser`

```
$ ./retweets-parser -h
  Usage of ./retweets-parser:
    -tweetId uint
        Tweet id for load its retweeters (default 1)
```

App `outgoing-waves-tx-recipients`

```
$ ./outgoing-waves-tx-recipients -h
Usage of ./outgoing-waves-tx-recipients:
  -fromTs int
        Filter transaction from timestamp in millis, optional
  -senderAddress string
        Sender address for filter the outcome transactions
  -tokenId string
        Token id for filter the outcome transactions
```

Example

Plain text output:

```
$ unzip release-linux-*.zip -d vrf-lottery
$ ./create-keys
Private key: GeSYYp42k3xtub6YycKFdWhX8gCR2byM5221VoPKC9Kd
Public key: rrKDRqKSZg5E9MxXyd6vgqv8ag1a2ZEorCtqTCVntCw
$ ./pick-winner -participantsFile ../../participants.txt -blockHeight
100 -pickN 5 -privateKey GeSYYp42k3xtub6YycKFdWhX8gCR2byM5221VoPKC9Kd
Provable lottery data was saved to file
'participants_and_100_block_signature.txt'
message:
["00amethyst00","01_youtube","0732917101","087745562868","15june","1ask_la

4p8zPAtyT18sWBewjUzhCDfTn5PWT9wwuJYJkG4qQTcGxoPAcHRCgnNWoRdYntu2dmjgf1ThX4

proof (base58):
kXpq4sD88nfAncTE7mxEDoeiPhAUsps3F7oUr7ZXVCkgQ5h3TUyHYvoLibL53s5xVc8dExkHZS

vrf bytes (base58): 847rcLh5JvHkaQ5HeWQyynYQyqD29vKaC8zr3P8vT6CU
winners are participants: [lheybralston untitled_av doankmarley
_risingdigital megoing]
$ ./verify-winner -provableFile participants_and_100_block_signature.txt
-blockHeight 100 -proof
kXpq4sD88nfAncTE7mxEDoeiPhAUsps3F7oUr7ZXVCkgQ5h3TUyHYvoLibL53s5xVc8dExkHZS
 -publicKey rrKDRqKSZg5E9MxXyd6vgqv8ag1a2ZEorCtqTCVntCw -pickN 5
message:
["00amethyst00","01_youtube","0732917101","087745562868","15june","1ask_la

4p8zPAtyT18sWBewjUzhCDfTn5PWT9wwuJYJkG4qQTcGxoPAcHRCgnNWoRdYntu2dmjgf1ThX4

proof (base58):
kXpq4sD88nfAncTE7mxEDoeiPhAUsps3F7oUr7ZXVCkgQ5h3TUyHYvoLibL53s5xVc8dExkHZS

vrf bytes (base58): 847rcLh5JvHkaQ5HeWQyynYQyqD29vKaC8zr3P8vT6CU
winners are participants: [lheybralston untitled_av doankmarley
_risingdigital megoing]
```

JSON output:

```
$ ./create-keys -json
{"private_key":"DRX1TGjTTLduUqj4VuwMugJxQH71ExJ8x3Kbdvh7kmDD","public_key"

➜  mac git:(master) ✗ ./verify-winner -provableFile
participants_and_100_block_signature.txt -blockHeight 100 -proof
kXpq4sD88nfAncTE7mxEDoeiPhAUsps3F7oUr7ZXVCkgQ5h3TUyHYvoLibL53s5xVc8dExkHZS
 -publicKey rrKDRqKSZg5E9MxXyd6vgqv8ag1a2ZEorCtqTCVntCw -pickN 3 -json
$ ./pick-winner -participantsFile ../../participants.txt -blockHeight
150 -pickN 3 -privateKey DRX1TGjTTLduUqj4VuwMugJxQH71ExJ8x3Kbdvh7kmDD -
json
{"winners":
["prettyktm","MarceloAvalos","PMYRS"],"proof":"2BJEFqEAfY9aHiFLvYMoPzMXeUh
[\"00amethyst00\",\"01_youtube\",\"0732917101\",\"087745562868\",\"15june\

➜  mac git:(master) ✗ ./pick-winner -participantsFile
../../participants.txt -blockHeight 150 -pickN 3 -privateKey
DRX1TGjTTLduUqj4VuwMugJxQH71ExJ8x3Kbdvh7kmDD -json
➜  mac git:(master) ✗ ./verify-winner -provableFile
participants_and_150_block_signature.txt -blockHeight 100 -proof
2BJEFqEAfY9aHiFLvYMoPzMXeUhztJCZs5VaVx1k4y4cQRMgpnW6W5pNuWeNUbmW6ewwRXrxdo
 -publicKey CqRguiUryNigwb17KEExeg2Ufn5jQnFvqVfnmqqivZ4E -pickN 3
Provable file contains different block signature at height 100: expected
'4p8zPAtyT18sWBewjUzhCDfTn5PWT9wwuJYJkG4qQTcGxoPAcHRCgnNWoRdYntu2dmjgf1ThX
 got
'TnPGMAhF8UiEXXDx23Dv4oZVX1BU2Y2LuTVsDjcJzo7HWNjXKitYqTPoxtbrGXCG3JsuxjLua

$ ./verify-winner -provableFile participants_and_150_block_signature.txt
-blockHeight 150 -proof
2BJEFqEAfY9aHiFLvYMoPzMXeUhztJCZs5VaVx1k4y4cQRMgpnW6W5pNuWeNUbmW6ewwRXrxdo
 -publicKey CqRguiUryNigwb17KEExeg2Ufn5jQnFvqVfnmqqivZ4E -pickN 3 -json
{"winners":
["prettyktm","MarceloAvalos","PMYRS"],"proof":"2BJEFqEAfY9aHiFLvYMoPzMXeUh
[\"00amethyst00\",\"01_youtube\",\"0732917101\",\"087745562868\",\"15june\
```