

# D-voting backend testing



Guanyu Zhang



Giovanni Igowa

- Introduction & Goals
- Quick overview of the current system
- Specifications and Test framework
- Results & Evaluation
- Proposed solutions
- Conclusion and future works

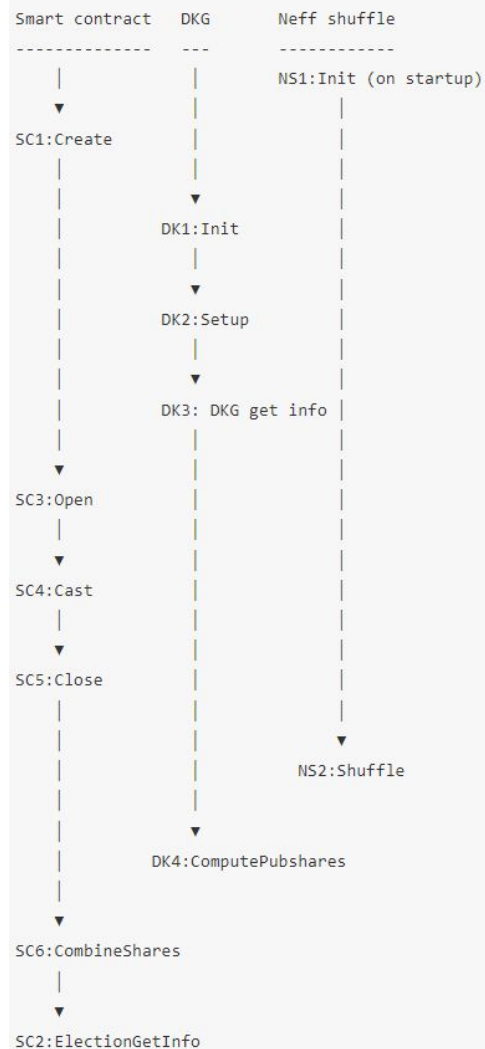
Goal: Providing a production-ready d-voting system

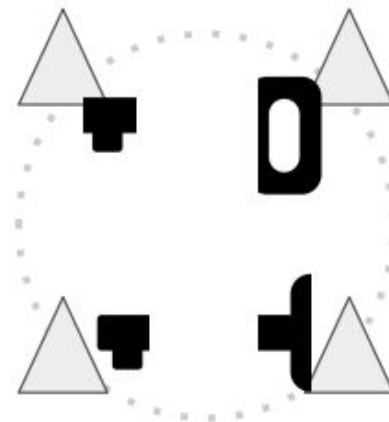
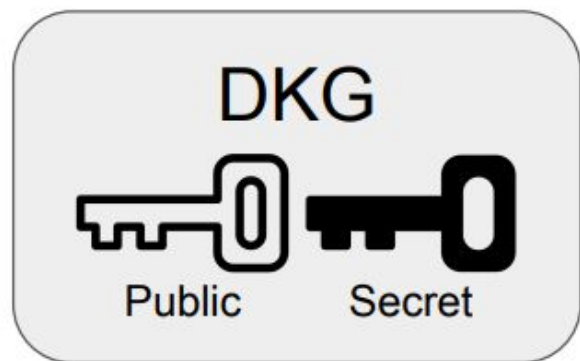
We need:

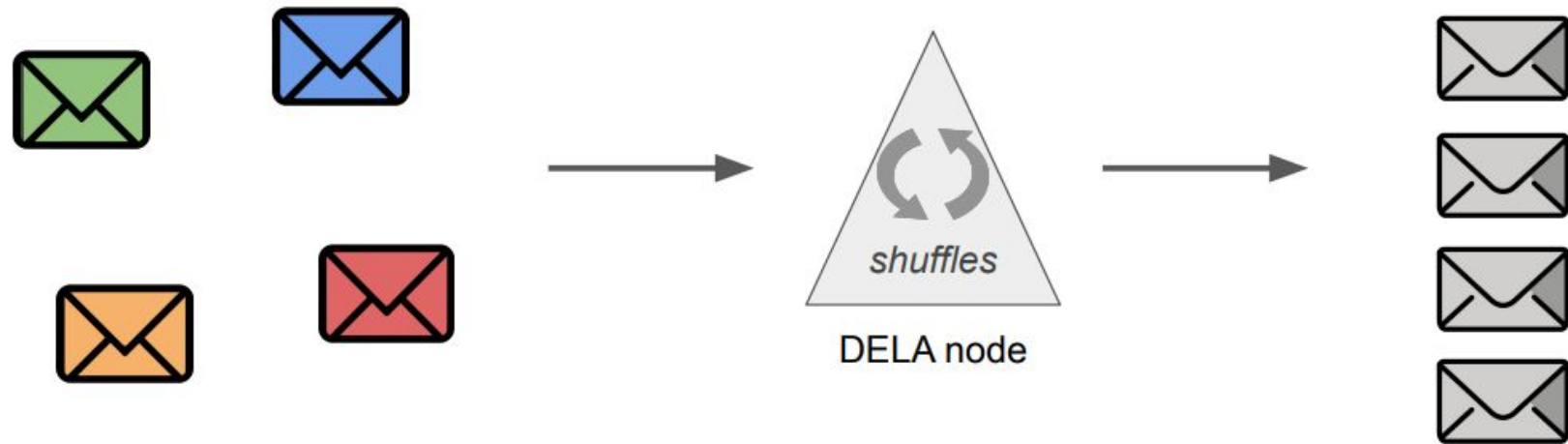
- Complete the prototype
- Define specifications of system
- Create testing framework and protocol
- Find limitations and propose solutions.

# System overview

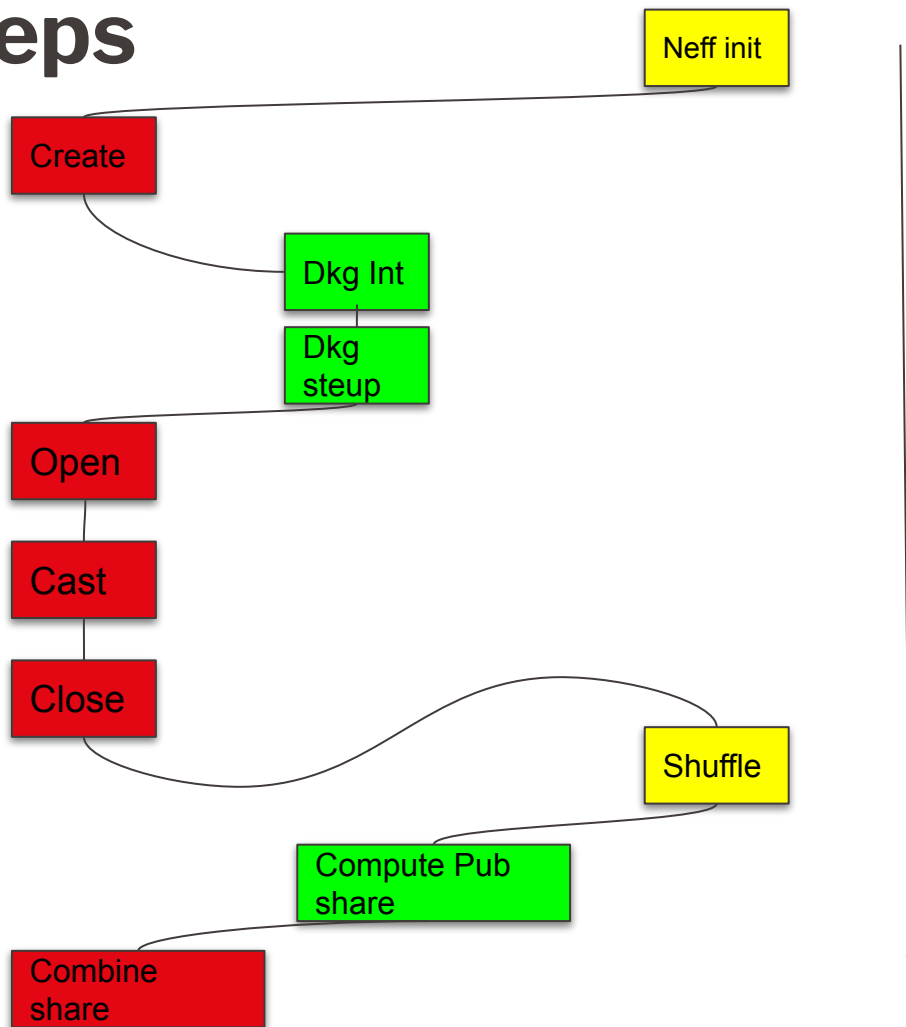
- DKG
- Shuffle
- Smart contract
- Election step







# Election steps



- Correctness
  - Voter privacy
  - Decryption correctness
- Scalability: ~ 10 000 voters and dozens of nodes
- Performance: setup + shuffle + decrypt  $\approx O(\text{day})$
- Resilience to node failure: theoretically supports  $\frac{1}{3}$  - 1 nodes down
  - continue to work / resume after nodes back



- Consistent results by launching tests a large number of times
- Store all log files and analyse them
- Deployment platform: locally or on docker containers

Why docker containers?

- slower (so more realistic) network communication
- ease to shut down and restart a node
- need less resources to deploy than VM
- industrial standard deployment tool, close to production environment

Test entrypoint: Scenario Test

- # simultaneous elections
- # nodes
- # votes
- Node failure
- Node restart
- Time measure
- Log generation
- Votes content comparaison

# Result and Evaluation

- Correctness
- Scalability
- Resilience
- Performance

# Correctness and scalability

- Increase number of nodes: from 3 to 25
- Increase number of ballots : from 3 to 200

→ What is the system behavior ?

Number of nodes n	Success rate test 15 attempts	Time (in seconds)
3	100%	59.93s
4	100%	66.02s
5	100%	73.37s
6	100%	92.81
7	100%	98.78
8	100%	115.97s
9	100%	135.77s
10	100%	127.89s
12	100%	169.73s
15	100 %	198.19s
20	FAIL timeout test == 10m %	600s
25	FAIL timeout test == 10m %	600s

**TABLE 3.1**

The evolution of scenario test success rate with the number of nodes (local)

Number of nodes n	Success rate test 15 attempts	Time (in seconds)
3	100%	62.80s
4	100%	66.70s
5	100%	76.18s
6	100%	77.19s
7	100%	88.89s
8	100%	86.52s
9	100%	113.18s
10	100%	116.28s
12	100%	123.48s
15	100 %	147.98s
20	FAIL timeout test == 10m	600s
25	FAIL timeout test == 10m	600s

**TABLE 3.2**

The evolution of scenario test success rate with the number of nodes (on docker)

```
cosipbft/proc.go:165 > view message refused error="invalid view:  
mismatch leader 9 != 13" addr=172.18.0.4:2001
```

→ “invalid view: mismatch leader”

# Correctness and scalability

- Increase number of nodes: from 3 to 25
- Increase number of ballots : from 3 to 200

→ What is the system behavior ?

# ballots n	Success rate test /Time (in seconds) 3 nodes	Success rate test /Time (in seconds) 5 nodes	Success rate test /Time (in seconds) 10 nodes
3	100% / 47.63s	100% / 59.93s	100% / 90.93s
10	100% / 67.91s	100% / 78.87	100% / 100.22s
25	100% / 86.95s	100% / 112.91s	100% / 183.29s
50	100% / 104.06s	100% / 154.64s	100% / 213.768854116
100	100% / 223.95s	100% / 303.51s	FAIL timeout test == 10m / 600s
150	100% / 360.40s	FAIL timeout test == 10m / 600s	FAIL timeout test == 10m / 600s
200	FAIL timeout test == 10m / 600s	FAIL timeout test == 10m / 600s	FAIL timeout test == 10m / 600s

# ballots n	Success rate test /Time (in seconds) 3 nodes	Success rate test /Time (in seconds) 5 nodes	Success rate test /Time (in seconds) 10 nodes
3	100% / 71.72s	100% / 74.26s	100% / 154.91s
10	100% / 65.48s	100% / 95.88s	100% / 191.17s
25	100% / 107.63s	100% / 160.25s	100% / 197.15s
50	100% / 178.39s	100% / 294.92s	FAIL timeout test == 10m / 600s
100	100% / 331.22s	FAIL timeout test == 10m / 600s	FAIL timeout test == 10m / 600s
150	100% / 367.62s	FAIL timeout test == 10m / 600s	FAIL timeout test == 10m / 600s
200	FAIL timeout test == 10m / 600s	FAIL timeout test == 10m / 600s	FAIL timeout test == 10m / 600s

TABLE 3.4

The evolution of scenario test success rate with the number of ballots (on docker)

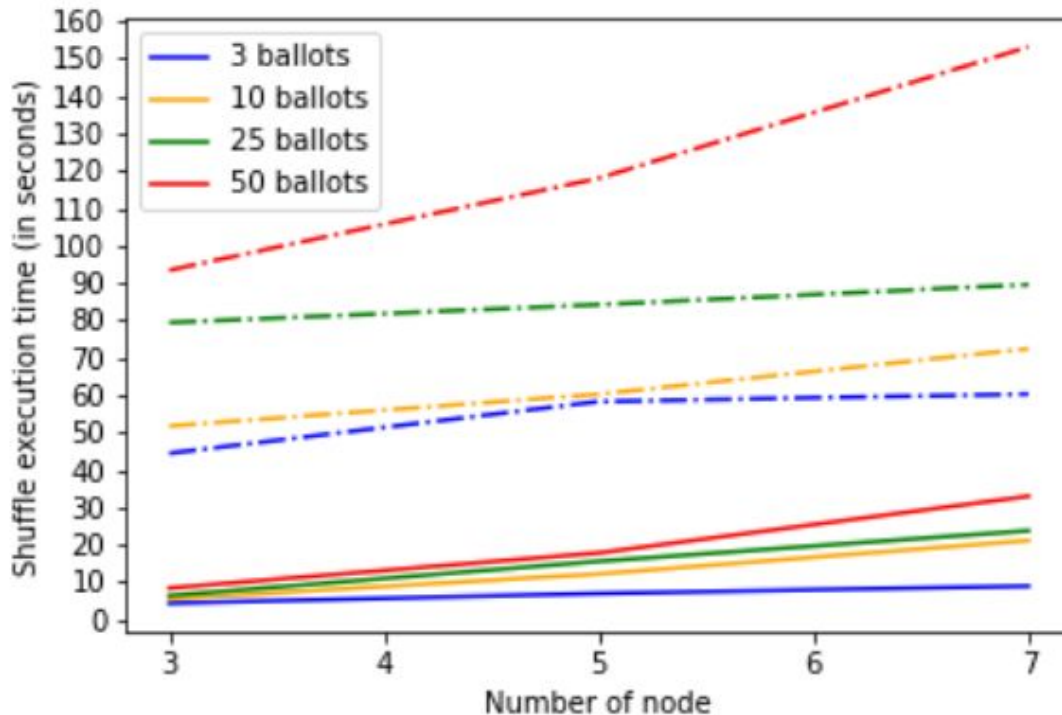


# Performance test

Goal: can election be run in a reasonable time?

Critical step: Shuffling

Extrapolation: 10 000  
votes on 14 nodes on  
docker takes 16.6 hours



# Resilience test

Goal: verify theoretical tolerant threshold  $\frac{1}{3}$  - 1 node failure

Critical steps:

1. setup / voting 
2. shuffling 
3. decryption 

⇒ kill nodes before shuffling

Number of nodes n	Shuffle success rate 15 attempts	Test success rate 15 attempts
4	100%	100%
5	100%	100%
6	100%	100%
7	100%	100%
8	100%	100%
9	100%	100%
10	100%	100%
11	100%	100%
12	100%	100%
13	100%	100%

locally, kill 1 node before shuffling

Number of nodes n	Success rate shuffle 15 attempts	Success rate test 15 attempts
7	100%	100%
8	100%	100%
9	100%	100%
10	100%	100%
11	100%	100%
12	100%	100%
13	100%	100%
14	100%	100%
15	100%	100%
16	100%	100%

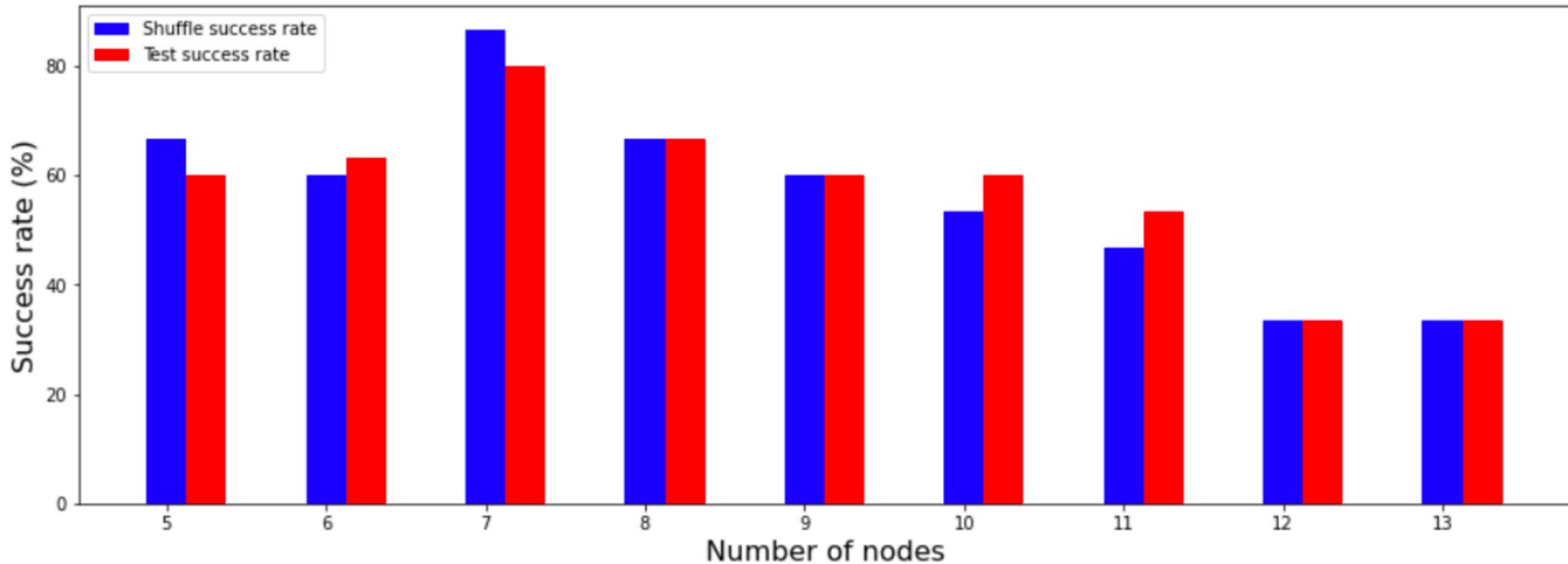
locally, kill 2 nodes before shuffling

Number of node n	Success rate shuffle 15 attempts	Success rate test 15 attempts
3	0%	0%
5	66.7% (10/15)	60% (9/15)
6	60% (9/15)	63.3% (8/15)
7	86.7% (13/15)	80 % (12/15)
8	66.7% (10/15)	66.7% (10/15)
9	60% (9/15)	60% (9/15)
10	53.3% (8/15)	60% (9/15)
11	46.7% (7/15)	53.3% (8/15)
12	33.3% (5/15)	33.3% (5/15)
13	33.3%( 5/15)	33.3% (5/15)

Docker, kill 1 node before shuffling

Number of node n	Shuffle step success rate 5 attempts
7	(0/5)
8	(0/5)
9	(0/5)
10	(0/5)
11	(0/5)
12	(0/5)
13	(0/5)
14	(0/5)

Docker, kill 2 nodes before shuffling



Number of node n	Election success rate 5 attempts
7	(5/5)
8	(5/5)
9	(5/5)
10	(5/5)
11	(5/5)
12	(5/5)
13	(5/5)
14	(5/5)

Docker, kill 2 nodes and restart those nodes

# Proposed solutions

Targeted problems:

1. Connection refused error
2. Threshold not reached error
3. View change error

⇒




Solutions

1. Shuffling retry: function of node number
2. Timeout elect new leader: function of node number

Exact hyperparameters: to be found by grid search

# Conclusion and Future works

D-voting system is partially validated in production environment

-  Able to handle large number of votes in a reasonable time
  -  Able to be deployed on a dozen of nodes
  -  Fault tolerant not validated in a close to production environment
- ...but election terminates correctly when failed nodes are back

Further steps:

1. Find the right hyper parameters of waiting times for shuffling and leader election
2. Find other efficient ways to handle communication overhead
3. Deploy on multiple machines
4. Compare our system performance with other e-voting systems