
Κατανεμημένα Συστήματα

Εξαμηνιαία Εργασία

BlockChat

[To L^AT_EX Project **εδώ**]

Ομάδα 29

Δημήτριος-Δαυίδ Γεροκωνσταντής (AM : 03119209)
Αθανάσιος Τσουκλείδης-Καρυδάκης (AM : 03119009)
Ιωάννης Καραυγουσής (AM : 03119847)

Εαρινό 2024



Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή ΗΜΜΥ

Contents

1	Δομή και λειτουργικότητα του BlockChat	3
1.1	Δομή-Αρχιτεκτονική του BlockChat	3
1.2	Λειτουργικότητα	5
1.3	Διαχείριση Stake	8
2	CLI και Frontend	8
2.1	CLI	8
2.2	Frontend	9
3	Μετρήσεις και αποτελέσματα	11
3.1	5 clients - block capacity 5, 10, 20 - ίδια stakes για όλους	11
3.2	Κλιμακωσιμότητα του Συστήματος	13
3.3	Δικαιοσύνη	13

Εισαγωγή

Σκοπός αυτής της εργασίας είναι η σχεδίαση και υλοποίηση μιας εφαρμογής βασισμένης στην τεχνολογία του Blockchain. Συγκεκριμένα πρόκειται για ένα κατανεμημένο σύστημα μέσω του οποίου διάφοροι χρήστες μπορούν να επιτελέσουν δύο λειτουργίες σε μορφή transactions :

1. Ανταλλαγή μηνυμάτων
2. Μεταφορά χρημάτων

Σκοπός είναι να επιτευχθεί η ορθή και συνεπής λειτουργία του συστήματος κάτω από πραγματικές συνθήκες χρήσης από πολλαπλούς participants ταυτόχρονα, όπου το σύστημα επιβαρύνεται με πολλά παράλληλα transactions. Προφανώς, σκοπός είναι και η υλοποίηση των βασικών δομικών χαρακτηριστικών της τεχνολογίας του blockchain που αφορούν την αυτόνομη αλλά και συνεπή λειτουργία των κόμβων με όλους τους απαραίτητους μηχανισμούς επικύρωσης των transactions και των blocks που δημιουργούνται. Η παρούσα αναφορά χωρίζεται στα παρακάτω τρία κεφάλαια :

1. **Κεφάλαιο 1^ο**: Παρουσίαση της δομής του κατανεμημένου συστήματος και των βασικών λειτουργιών του BlockChat. Αναφορά σε επιπλέον χαρακτηριστικά που υλοποιήθηκαν με σκοπό την ορθότερη λειτουργία του κατανεμημένου συστήματος.
2. **Κεφάλαιο 2^ο**: Παρουσίαση του Frontend και του CLI.
3. **Κεφάλαιο 3^ο**: Παρουσίαση των αποτελεσμάτων των μετρήσεων που πραγματοποιήθηκαν με χρήση VMs από την cloud υπηρεσία του okeanos-knossos.

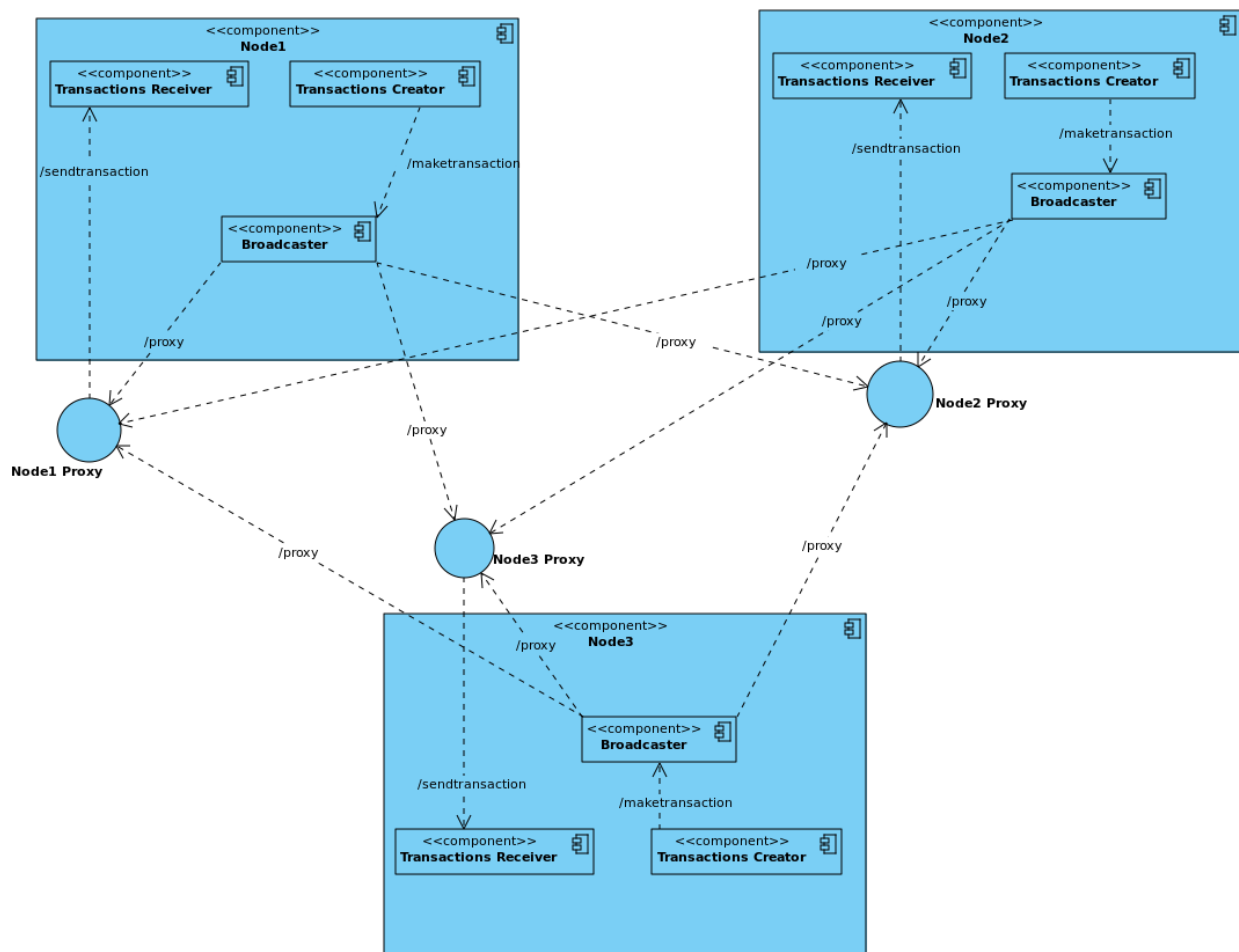
1 Δομή και λειτουργικότητα του BlockChat

Στο κεφάλαιο αυτό παρουσιάζεται αρχικά η αρχιτεκτονική δομή του συστήματός μας, τα εργαλεία που χρησιμοποιήθηκαν για την υλοποίησή του καθώς και οι λειτουργικότητές του. Να σημειωθεί ότι για την υλοποίηση του BlockChat backend χρησιμοποιήθηκε η python flask.

1.1 Δομή-Αρχιτεκτονική του BlockChat

Η δομή του κατανεμημένου συστήματος που υλοποιούμε φαίνεται στην εικόνα 1, όπου παρουσιάζονται τα βασικά δομικά χαρακτηριστικά των κόμβων του BlockChat καθώς και του τρόπου επικοινωνίας μεταξύ τους. Όπως φαίνεται στο σχήμα για 3 κόμβους συνολικά, κάθε participant του blockchat περιέχει:

1. ένα component που διαχειρίζεται τα εισερχόμενα transactions (Transactions Receiver)
2. ένα component (Transactions Creator) που δημιουργεί και προωθεί νέα transactions σε ένα component (Broadcaster) που υλοποιεί την broadcast αποστολή τους στους άλλους συμμετέχοντες
3. Έναν μηχανισμό proxy ο οποίος διαχειρίζεται εισερχόμενα requests, δηλαδή εισερχόμενα transactions.



Εικόνα 1: Δομικά χαρακτηριστικά του BlockChat

Ο τρόπος λειτουργίας των πρώτων δύο components που υλοποιούν ουσιαστικά την λογική του BlockChat θα παρουσιαστεί στην συνέχεια. Ως προς τη δομή του συστήματος, ενδιαφέρον έχει η παρουσίαση της λειτουργικότητας του proxy. Η ουσία έγκειται στον πλήρη διαχωρισμό (decoupling) της υλοποίησης της λογικής του BlockChat από την επικοινωνία μεταξύ των κόμβων. Έτσι, κάθε κόμβος blockchat επιφορτίζεται μόνο με την διαχείριση των transactions και του blockchain, ενώ ο proxy είναι αυτός που αναλαμβάνει να δέχεται εισερχόμενα transactions από τους υπόλοιπους κόμβους και να τα προωθεί στα αντίστοιχα components του κόμβου με συγκεκριμένο ρυθμό και σειριακά. Πρακτικά, αυτό που συμβαίνει είναι ότι όταν ένας κόμβος δημιουργεί ένα transaction και το προωθεί σε όλους τους άλλους κόμβους, στην πραγματικότητα αυτό το transaction λαμβάνεται από τον proxy κάθε μηχανήματος (όπως και από τον δικό μας proxy). Έτσι τόσο για τα transactions που δημιουργεί ένας κόμβος όσο και για αυτά που λαμβάνει από τους υπόλοιπους κόμβους, υπάρχει ο proxy ώστε όλα αυτά τα transactions να τα αποθηκεύει σε μια ουρά και να τα προωθεί αυτός στο εσωτερικό του κόμβου προκαλώντας την λειτουργία της λογικής του blockchain. Όπως γίνεται αντιληπτό, για την διαχείριση των σύγχρονων transactions (υπενθυμίζεται ότι στο συστήμα μας, δημιουργούνται παράλληλα transactions από όλους τους κόμβους και κάθε κόμβος οφείλει να ενημερώνει την κατάστασή του τόσο βάσει των δικών του transactions όσο και βάσει των transactions των υπολοίπων), ακολουθήσαμε τη λογική της σειριοποίησής τους. Όλα τα transactions που συσ-

ωρεούνται στην ουρά του proxy, προωθούνται σειριακά στο σύστημα και εκτελούνται ατομικά. Κάθε επόμενο transaction προωθείται στο σύστημά μας αφότου έχει πλήρως εξυπηρετηθεί το προηγούμενο transaction. Με αυτό τον τρόπο το state κάθε κόμβου παραμένει συνεπές και ορθό. Ωστόσο, αυτή η λογική σειριοποίησης απαιτούσε και κάποιες επιπλέον προσθήκες στην υλοποίηση του BlockChat για λόγους συγχρονισμού και επιβολής συγκεκριμένης σειράς με την οποία εκτελούνται οι επιμέρους λειτουργίες. Για παράδειγμα όταν ένα block γίνεται mine και ο validator πρέπει να κάνει broadcast το block του, χρειάστηκε να εξασφαλιστεί ότι έχει ολοκληρωθεί το broadcast προτού ο validator αρχίζει να δέχεται νέα transactions, διαφορετικά θα υπήρχε ασυνέπεια σε μεταβλητές κατάστασης που διατηρεί ο εν λόγω κόμβος. Ένα δεύτερο παράδειγμα τέτοιου χειρισμού είναι ότι έπρεπε να διασφαλιστεί ότι πρώτα θα ενημερωθεί το state ενός κόμβου με βάση το τελευταίο transaction ενός block και μετά θα γίνει η αναίρεση του state και ο επανυπολογισμός του βάσει του block που στέλνει ο validator. Διαφορετικά, επίσης θα υπήρχε έλλειψη ορθότητας (η οποία παρατηρήθηκε χωρίς αυτή την παρέμβαση). Τέλος, σημειώνουμε ότι η επικοινωνία μεταξύ των κόμβων γίνεται μέσω REST API που δημιουργήθηκε με χρήση της βιβλιοθήκης flask της python.

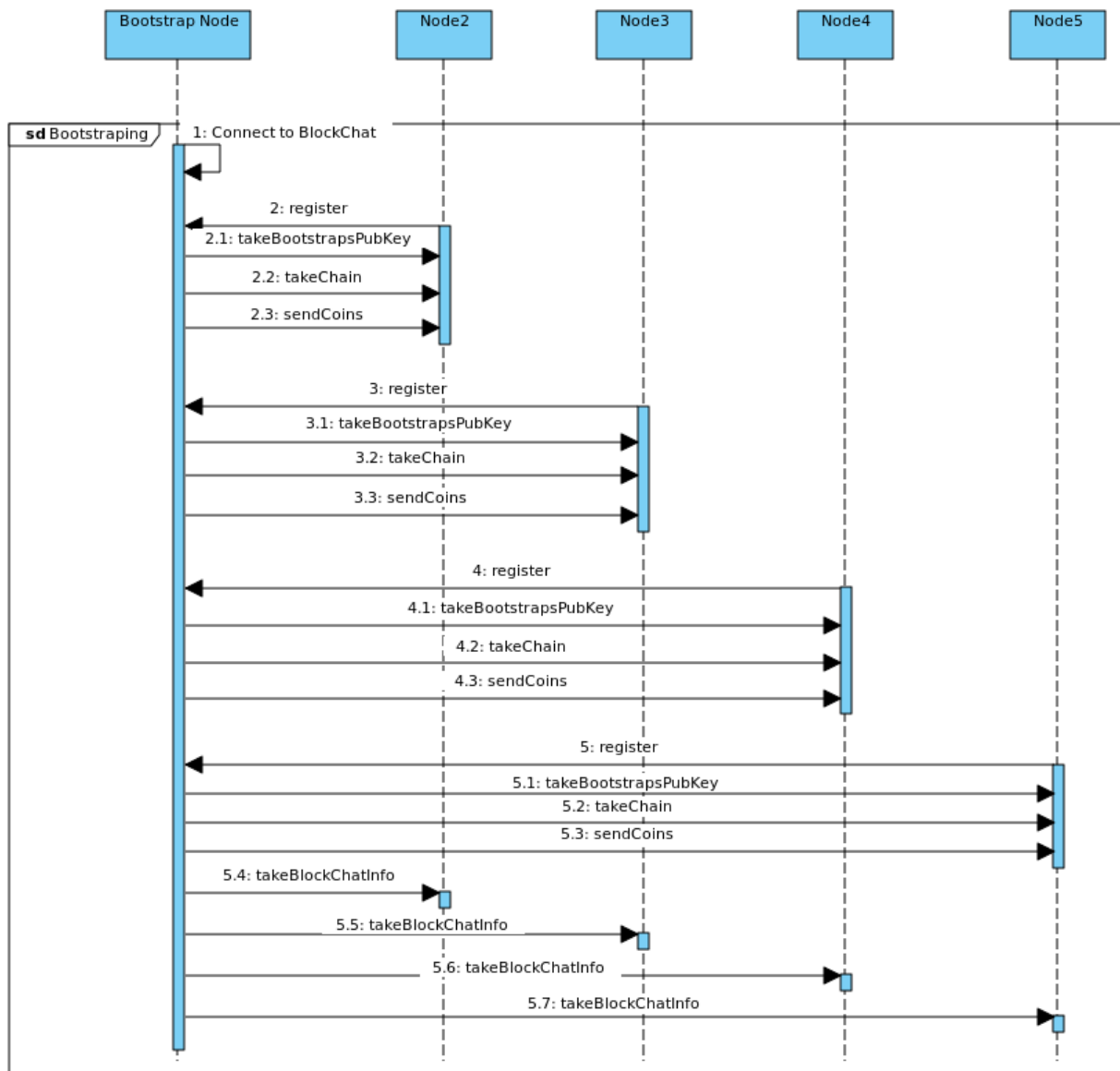
1.2 Λειτουργικότητα

Στο BlockChat θεωρούμε ότι υπάρχει ένας κόμβος, ο bootstrapping node, ο οποίος ενώ είναι ισότιμος κόμβος με τους υπόλοιπους, κατά την αρχικοποίηση του συστήματος αναλαμβάνει τις εξής βασικές λειτουργίες:

1. Πρόκειται για έναν κόμβο με γνωστή IP σε όλους τους υπόλοιπους. Ο bootstrapping node οφείλει να δέχεται αιτήματα για register από τους υπόλοιπους participants ώστε να τους εισάγει επιτυχώς στο blockchat κατά τη διαδικασία αρχικοποίησης. Σε κάθε νέο κόμβο που επικοινωνεί με αυτόν, αναλαμβάνει να τον ενημερώσει για το public key του, να του δώσει το ήδη υπάρχον Blockchain, να του αναθέσει ένα id και να του μεταφέρει το αρχικό χρηματικό ποσό των 1000 coins.
2. Όταν και ο τελευταίος κόμβος εισάγεται στο σύστημα, ο bootstrapping node αφού του μεταφέρει όσα και πριν, φροντίζει να ενημερώσει όλους τους participants με πληροφορίες για όλους τους υπόλοιπους participants ώστε μετά από αυτό το σημείο όλοι να γνωρίζουν όλους τους υπόλοιπους και να μπορούν να επικοινωνούν μεταξύ τους.

Αυτή η διαδικασία του bootstrapping φαίνεται σχηματικά στην εικόνα 2 όπου έχει σχεδιαστεί ένα sequence διάγραμμα.

Για λόγους αφενός διευκόλυνσης και αφετέρου συνέπειας με τους περιορισμούς της εκφώνησης, αποφασίστηκε να μην επιτρέπεται στους κόμβους να στέλνουν μεταξύ τους transactions προτού εισαχθούν όλοι οι κόμβοι στο BlockChat. Δεδομένου ότι όλοι οι κόμβοι γνωρίζουν τους υπόλοιπους μόνο στο τέλος του bootstrapping, η broadcast αποστολή transactions σε κόμβους που ακόμη δεν είναι γνωστοί δημιουργεί πρόβλημα στην ενημέρωση του state καθώς οι κόμβοι που λαμβάνουν τέτοια transactions δεν γνωρίζουν ενδεχομένως τους εμπλεκόμενους σε αυτά. Επιπλέον, ακόμη κι αν υλοποιούσαμε κάτι τέτοιο, transactions αυτού του τύπου θα μπορούσαν να στέλνονται μόνο στον bootstrapping κόμβο, καθώς πριν την εισαγωγή όλων των κόμβων, κάθε κόμβος γνωρίζει μόνο τον bootstrapping και μόνο με αυτόν θα μπορούσε ενδεχομένως να ανταλλάξει transactions. Τέλος, ως προς το bootstrapping αξίζει να σημειωθεί ότι ο bootstrapping node είναι έτσι υλοποιημένος ώστε η διαδικασία της αποστολής των 1000 coins, του chain

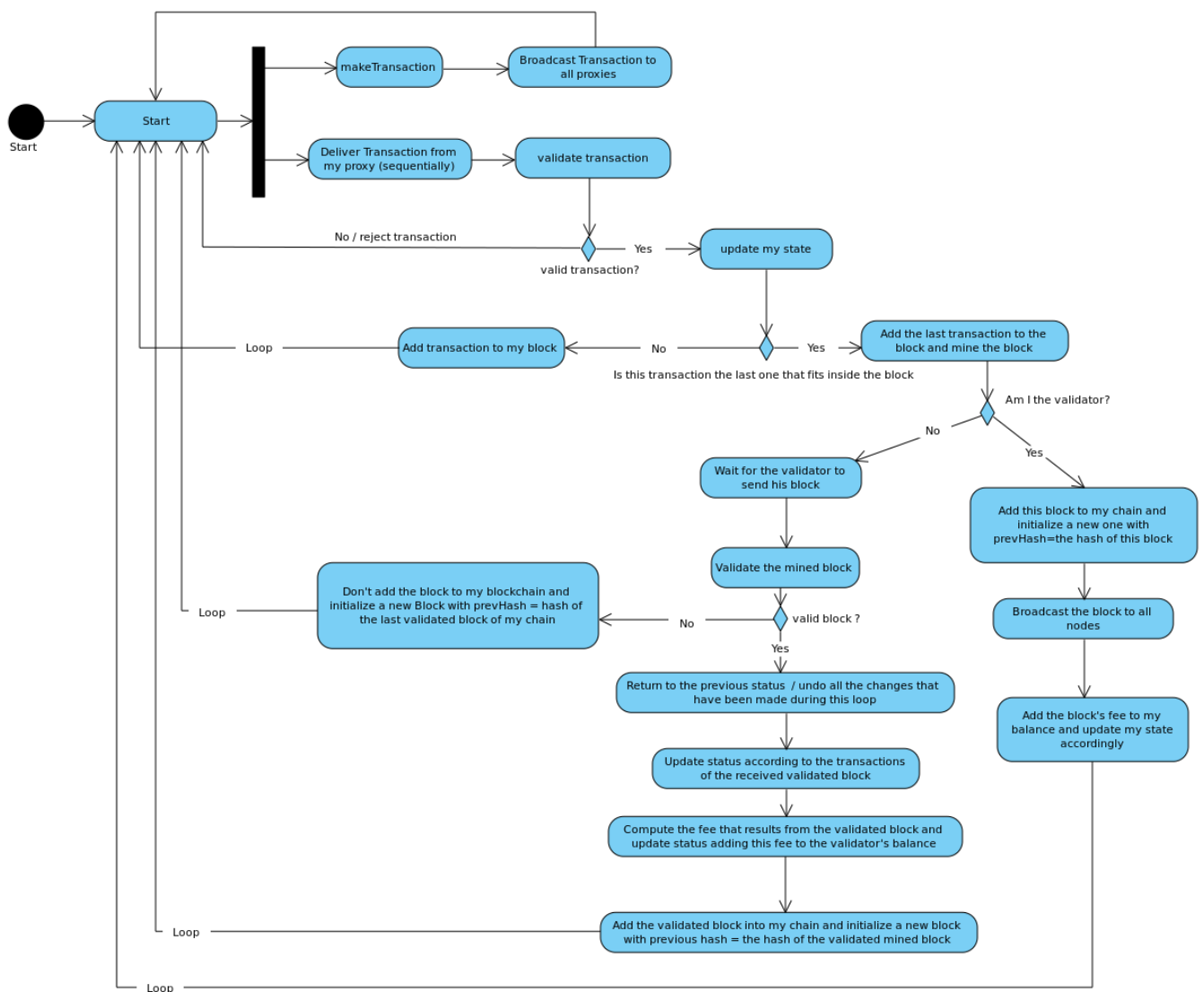


Εικόνα 2: Σχηματική αναπαράσταση του Bootstrapping για blockchain με 5 κόμβους

και των συνολικών πληροφοριών στο τέλος να γίνεται από ένα άλλο νήμα (χρήση της βιβλιοθήκης threading στην python). Στη συνέχεια, παρουσιάζεται η βασική λειτουργικότητα κάθε κόμβου του blockchain. Αυτή η λειτουργικότητα παρουσιάζεται αρκετά αναλυτικά στο activity diagram της εικόνας 3. Συγκεκριμένα κάθε κόμβος ταυτόχρονα δημιουργεί νέα transactions και δέχεται άλλα transactions που δημιουργούνται από άλλους κόμβους. Η διαδικασία από εκεί και έπειτα περιγράφεται περιεκτικά στο διάγραμμα αλλά αξίζει να σημειωθούν τα εξής :

- Όταν ένας κόμβος δέχεται ένα block από τον validator, το επικυρώνει. Αυτή η διαδικασία ενώ θα μπορούσε να γίνει αρκετά πλήρης και περίπλοκη, ακολουθήσαμε την υπόδειξη της εκφώνησης σύμφωνα με την οποία το validation αφορά την επαλήθευση του previous hash καθώς και του αποστολέα (αν πράγματι είναι ο εκλεγμένος από όλους validator). Δηλαδή δεν γίνεται κάποιο parse του block για επαλήθευση των transactions που περιλαμβάνει.

- Κάθε κόμβος διατηρεί ένα state για την κατάσταση του blockchat (π.χ. τα balances όλων των κόμβων, τα επόμενα nonces που περιμένει να λάβει από κάθε κόμβο, τα stakes τους κλπ). Όταν ένα block γίνεται mine, ο validator απλώς διατηρεί το state του καθώς τελικά αυτό θα επικρατήσει σε όλους. Ωστόσο, μέχρι να γίνει mine αυτό το block, στο μεταξύ οι υπόλοιποι κόμβοι που χτίζουν το δικό τους block ενδέχεται να έχουν διαφορετική εντύπωση για το state του συστήματος. Ως μη validators, όλοι αυτοί οι κόμβοι οφείλουν να υιοθετήσουν την άποψη του validator. Συγκεκριμένα, από κάθε κόμβο διατηρούνται κάποια checkpoints (ουσιαστικά κατά την μετάβαση σε νέο block) με το state του, έτσι ώστε αν δεν είναι validator, να επιστρέψει στο προηγούμενο state του (δηλαδή να κάνει rollback τις αλλαγές που έχει προκαλέσει σε αυτό) και να κάνει rerun όλα τα transactions που περιέχονται στο received validated block.



Εικόνα 3: Λειτουργικότητα ενός BlockChat κόμβου

1.3 Διαχείριση Stake

Ως προς τη λειτουργικότητα του συστήματος, αξίζει να αναφερθεί ο τρόπος διαχείρισης του stake των κόμβων. Δεδομένου ότι η οποιαδήποτε αλλαγή σε stake αποτελεί ένα transaction, θεωρήθηκε χρήσιμο και δίκαιο η αλλαγή αυτή να γίνει εμφανής στους κόμβους μόνο μετά το mining του block που περιλαμβάνει το εν λόγω transaction. Συνεπώς, αν σε ένα block εισαχθεί ένα transaction αλλαγής stake, η χρήση του νέου stake θα χρησιμοποιηθεί για το mining του επόμενου και όχι του παρόντος block. Ωστόσο, ακόμα και με αυτή την υλοποίηση υπάρχει ακόμη πρόβλημα. Αν στο μέσο ενός block κάποιος κόμβος με balance 500 αποφασίζει να εισαγάγει ένα transaction αλλαγής stake σε 400, οι υπόλοιποι κόμβοι που λάβουν με broadcast τρόπο αυτό το transaction προφανώς θα το επικυρώσουν. Ωστόσο, έστω ότι αυτός ο κόμβος προτού γεμίσει το συγκεκριμένο block εισάγει ένα νέο transaction μεταφορά 200 coins σε κάποιον άλλο. Δεδομένου ότι το stake 400 ακόμη δεν έχει ενημερώσει την κατάσταση των κόμβων, αυτό το transaction θα επικυρωθεί επίσης. Έτσι, ο εν λόγω κόμβος κατάφερε και να αυξήσει το stake του (αυξάνοντας την πιθανότητα να γίνει αργότερα validator) και κατάφερε να κάνει transactions που υπερβαίνουν αυτό το stake. Για την επίλυση αυτού του φαινομένου, οι κόμβοι (είτε όταν λαμβάνουν είτε όταν κάνουν broadcast ένα νέο block) εντοπίζουν τα stake transactions που περιλαμβάνονται σε αυτό και αφότου ενημερώσουν στο state τους με όλα τα υπόλοιπα transactions του block, στο τέλος ελέγχουν ξανά ότι το συγκεκριμένο transaction αλλαγής stake εξακολουθεί να είναι έγκυρο. Αν είναι έγκυρο, τότε πράγματι ενημερώνουν το stake του κόμβου, διαφορετικά δεν επικυρώνουν αυτή την αλλαγή.

2 CLI και Frontend

Στο κεφάλαιο αυτό, παρουσιάζεται το CLI και το Frontend που δημιουργήθηκαν στα πλαίσια της παρούσας εργασίας.

2.1 CLI

Μέσω του CLI, κάθε blockchain κόμβος έχει τη δυνατότητα να στέλνει transactions για messages ή coins, να βλέπει το balance του, να βλέπει το τελευταίο validated block του blockchain και τον validator του και να μεταβάλλει το stake του. Συγκεκριμένα, με χρήση της εντολής bchelp, βλέπουμε το description και τη χρήση κάθε τέτοιας εντολής:

```
1 This is a description of how to use this CLI
2
3 1st command
4 t <recipient address> <message>
5 Description: It sends the message <message> to the node with
   id=<recipient_address>
6
7 2nd command
8 c <recipient address> <amount>
9 Description: It sends <amount> coins to the node with
   id=<recipient_address>
10
11 3rd command
12 stake <amount>
13 Description: It updates the stake so as to be equal to <amount> coins
```



```

14
15 4th command
16 view
17 Description: It prints the last validated block of the blockchain and the
    validator of it
18
19 5th command
20 balance
21 Description: It prints the balance of the node

```

Το CLI δημιουργήθηκε με χρήση της βιβλιοθήκης click της python. Για κάθε αρχείο .py που δημιουργήθηκε για κάθε μια από τις εντολές του CLI, χρειάστηκε η εκτέλεση των παρακάτω:

```

1 sudo nano /usr/local/bin/<command_name>
2
3     Inside this file, write:
4     python3 <path_to_cli_py_file> "$@"
5
6 chmod +x /usr/local/bin/<command_name>

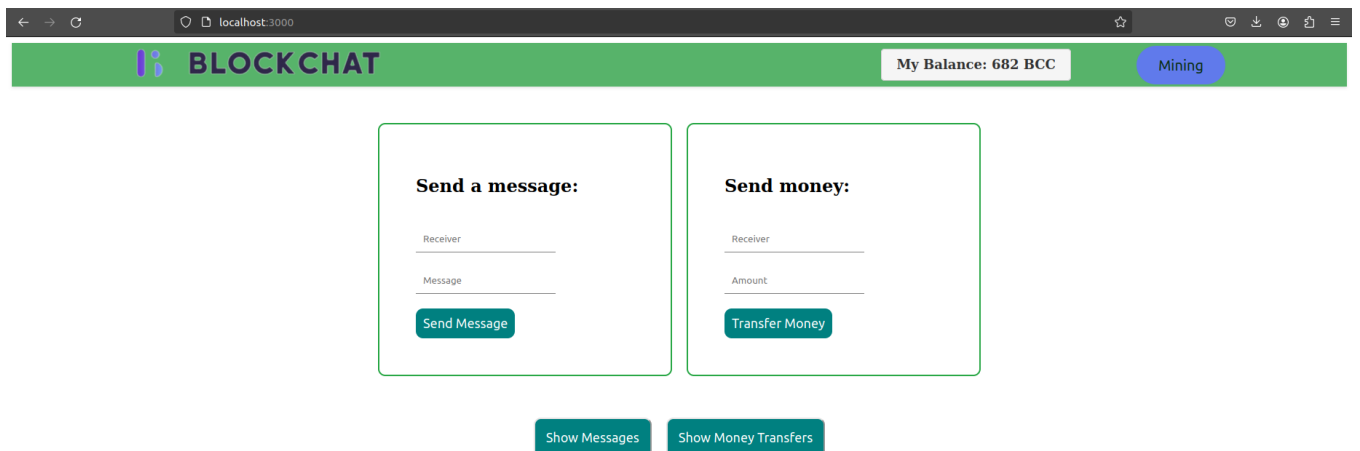
```

Επίσης με

```
1 <command_name> --help
```

μπορεί κανείς να διαβάσει πληροφορίες χρήσης της εκάστοτε εντολής.


2.2 Frontend




Εικόνα 4: Αρχική σελίδα Frontend

Στη συνέχεια, παρουσιάζεται το frontend (χρήση React.js) μέσω του οποίου κάθε κόμβος μπορεί να πραγματοποιήσει όλες τις βασικές λειτουργίες του BlockChat. Η αρχική σελίδα

φαίνεται στην εικόνα 4. Μέσω αυτής της σελίδας, ο χρήστης μπορεί να δημιουργήσει νέα message και coins transactions εισάγοντας το περιεχόμενο του transaction (coins ή message) και το id του receiver (για λόγους απλότητας, όχι το public key του). Επιπλέον μπορεί να δει το balance του. Επιλέγοντας Show Messages, προωθείται στην σελίδα που φαίνεται στην

<div><div> BLOCKCHAT</div><div>My Balance: 555.5 BCC</div><div>Mining</div></div>	
Outgoing Messages	
<div>Show Incoming Messages</div>	
Receiver	Message
-----BEGIN PUBLIC KEY----- MIIBJANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAE624dHsXXyDYGdzWWxFDY s5RNzKxpjSBjhQCTulOir3hEzKs5K5sBuilc8pkjLW/om64RzdHKwOc0e1F9UQP HJWCExHsHY5YHQGfGoghlf3Sez7HHJJNSLSrNoXrTb/aaJbKefAru/1n0b79K0 pWtMfC0iVqihagtgZZclmqfJoulouRdAs7ayPQTLpQOknI2xEu65kzXk3lw76bpj5uz8p6VWJ19gDv+A6GQl7Djsj59QITpudV5iljh3Gg6qNjollAlJxHuHrxlllI 8CeKyOr9mCLjma1RjsGvPG8sbI2AMjcjcf+uROZnyCLFTs0sKIWoHo0GpNfNDE6W 1wIDAQAB -----END PUBLIC KEY-----	In the beginning the Universe was created.
-----BEGIN PUBLIC KEY----- MIIBJANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAE624dHsXXyDYGdzWWxFDY s5RNzKxpjSBjhQCTulOir3hEzKs5K5sBuilc8pkjLW/om64RzdHKwOc0e1F9UQP HJWCExHsHY5YHQGfGoghlf3Sez7HHJJNSLSrNoXrTb/aaJbKefAru/1n0b79K0 pWtMfC0iVqihagtgZZclmqfJoulouRdAs7ayPQTLpQOknI2xEu65kzXk3lw76bpj5uz8p6VWJ19gDv+A6GQl7Djsj59QITpudV5iljh3Gg6qNjollAlJxHuHrxlllI 8CeKyOr9mCLjma1RjsGvPG8sbI2AMjcjcf+uROZnyCLFTs0sKIWoHo0GpNfNDE6W 1wIDAQAB -----END PUBLIC KEY-----	I'd far rather be happy than right any day.
-----BEGIN PUBLIC KEY----- MIIBJANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAE624dHsXXyDYGdzWWxFDY s5RNzKxpjSBjhQCTulOir3hEzKs5K5sBuilc8pkjLW/om64RzdHKwOc0e1F9UQP HJWCExHsHY5YHQGfGoghlf3Sez7HHJJNSLSrNoXrTb/aaJbKefAru/1n0b79K0 pWtMfC0iVqihagtgZZclmqfJoulouRdAs7ayPQTLpQOknI2xEu65kzXk3lw76bpj5uz8p6VWJ19gDv+A6GQl7Djsj59QITpudV5iljh3Gg6qNjollAlJxHuHrxlllI 8CeKyOr9mCLjma1RjsGvPG8sbI2AMjcjcf+uROZnyCLFTs0sKIWoHo0GpNfNDE6W 1wIDAQAB -----END PUBLIC KEY-----	I eventually had to go down to the cellar to find them.
-----BEGIN PUBLIC KEY----- MIIBJANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAE624dHsXXyDYGdzWWxFDY s5RNzKxpjSBjhQCTulOir3hEzKs5K5sBuilc8pkjLW/om64RzdHKwOc0e1F9UQP HJWCExHsHY5YHQGfGoghlf3Sez7HHJJNSLSrNoXrTb/aaJbKefAru/1n0b79K0 pWtMfC0iVqihagtgZZclmqfJoulouRdAs7ayPQTLpQOknI2xEu65kzXk3lw76bpj5uz8p6VWJ19gDv+A6GQl7Djsj59QITpudV5iljh3Gg6qNjollAlJxHuHrxlllI 8CeKyOr9mCLjma1RjsGvPG8sbI2AMjcjcf+uROZnyCLFTs0sKIWoHo0GpNfNDE6W 1wIDAQAB -----END PUBLIC KEY-----	So had the stairs.
-----BEGIN PUBLIC KEY----- MIIBJANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAE624dHsXXyDYGdzWWxFDY s5RNzKxpjSBjhQCTulOir3hEzKs5K5sBuilc8pkjLW/om64RzdHKwOc0e1F9UQP HJWCExHsHY5YHQGfGoghlf3Sez7HHJJNSLSrNoXrTb/aaJbKefAru/1n0b79K0 pWtMfC0iVqihagtgZZclmqfJoulouRdAs7ayPQTLpQOknI2xEu65kzXk3lw76bpj5uz8p6VWJ19gDv+A6GQl7Djsj59QITpudV5iljh3Gg6qNjollAlJxHuHrxlllI 8CeKyOr9mCLjma1RjsGvPG8sbI2AMjcjcf+uROZnyCLFTs0sKIWoHo0GpNfNDE6W 1wIDAQAB -----END PUBLIC KEY-----	Life!
-----BEGIN PUBLIC KEY----- MIIBJANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAE624dHsXXyDYGdzWWxFDY s5RNzKxpjSBjhQCTulOir3hEzKs5K5sBuilc8pkjLW/om64RzdHKwOc0e1F9UQP HJWCExHsHY5YHQGfGoghlf3Sez7HHJJNSLSrNoXrTb/aaJbKefAru/1n0b79K0 pWtMfC0iVqihagtgZZclmqfJoulouRdAs7ayPQTLpQOknI2xEu65kzXk3lw76bpj5uz8p6VWJ19gDv+A6GQl7Djsj59QITpudV5iljh3Gg6qNjollAlJxHuHrxlllI 8CeKyOr9mCLjma1RjsGvPG8sbI2AMjcjcf+uROZnyCLFTs0sKIWoHo0GpNfNDE6W 1wIDAQAB -----END PUBLIC KEY-----	Yes...!!!!?
-----BEGIN PUBLIC KEY----- MIIBJANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAE624dHsXXyDYGdzWWxFDY s5RNzKxpjSBjhQCTulOir3hEzKs5K5sBuilc8pkjLW/om64RzdHKwOc0e1F9UQP HJWCExHsHY5YHQGfGoghlf3Sez7HHJJNSLSrNoXrTb/aaJbKefAru/1n0b79K0 pWtMfC0iVqihagtgZZclmqfJoulouRdAs7ayPQTLpQOknI2xEu65kzXk3lw76bpj5uz8p6VWJ19gDv+A6GQl7Djsj59QITpudV5iljh3Gg6qNjollAlJxHuHrxlllI 8CeKyOr9mCLjma1RjsGvPG8sbI2AMjcjcf+uROZnyCLFTs0sKIWoHo0GpNfNDE6W 1wIDAQAB -----END PUBLIC KEY-----	The Answer to the Great Question...
-----BEGIN PUBLIC KEY----- MIIBJANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAE624dHsXXyDYGdzWWxFDY s5RNzKxpjSBjhQCTulOir3hEzKs5K5sBuilc8pkjLW/om64RzdHKwOc0e1F9UQP HJWCExHsHY5YHQGfGoghlf3Sez7HHJJNSLSrNoXrTb/aaJbKefAru/1n0b79K0 pWtMfC0iVqihagtgZZclmqfJoulouRdAs7ayPQTLpQOknI2xEu65kzXk3lw76bpj5uz8p6VWJ19gDv+A6GQl7Djsj59QITpudV5iljh3Gg6qNjollAlJxHuHrxlllI 8CeKyOr9mCLjma1RjsGvPG8sbI2AMjcjcf+uROZnyCLFTs0sKIWoHo0GpNfNDE6W 1wIDAQAB -----END PUBLIC KEY-----	To Everything?

Εικόνα 5: Outgoing Messages Page

<div><div> BLOCKCHAT</div><div>My Balance: 594.6 BCC</div><div>Mining</div></div>	
Outgoing Money Transfers	
<div>Show Incoming Money Transfers</div>	
Receiver	Amount
-----BEGIN PUBLIC KEY----- MIIBJANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAE624dHsXXyDYGdzWWxFDY s5RNzKxpjSBjhQCTulOir3hEzKs5K5sBuilc8pkjLW/om64RzdHKwOc0e1F9UQP HJWCExHsHY5YHQGfGoghlf3Sez7HHJJNSLSrNoXrTb/aaJbKefAru/1n0b79K0 pWtMfC0iVqihagtgZZclmqfJoulouRdAs7ayPQTLpQOknI2xEu65kzXk3lw76bpj5uz8p6VWJ19gDv+A6GQl7Djsj59QITpudV5iljh3Gg6qNjollAlJxHuHrxlllI 8CeKyOr9mCLjma1RjsGvPG8sbI2AMjcjcf+uROZnyCLFTs0sKIWoHo0GpNfNDE6W 1wIDAQAB -----END PUBLIC KEY-----	5000
-----BEGIN PUBLIC KEY----- MIIBJANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAE624dHsXXyDYGdzWWxFDY s5RNzKxpjSBjhQCTulOir3hEzKs5K5sBuilc8pkjLW/om64RzdHKwOc0e1F9UQP HJWCExHsHY5YHQGfGoghlf3Sez7HHJJNSLSrNoXrTb/aaJbKefAru/1n0b79K0 pWtMfC0iVqihagtgZZclmqfJoulouRdAs7ayPQTLpQOknI2xEu65kzXk3lw76bpj5uz8p6VWJ19gDv+A6GQl7Djsj59QITpudV5iljh3Gg6qNjollAlJxHuHrxlllI 8CeKyOr9mCLjma1RjsGvPG8sbI2AMjcjcf+uROZnyCLFTs0sKIWoHo0GpNfNDE6W 1wIDAQAB -----END PUBLIC KEY-----	1000
-----BEGIN PUBLIC KEY----- MIIBJANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAE624dHsXXyDYGdzWWxFDY s5RNzKxpjSBjhQCTulOir3hEzKs5K5sBuilc8pkjLW/om64RzdHKwOc0e1F9UQP HJWCExHsHY5YHQGfGoghlf3Sez7HHJJNSLSrNoXrTb/aaJbKefAru/1n0b79K0 pWtMfC0iVqihagtgZZclmqfJoulouRdAs7ayPQTLpQOknI2xEu65kzXk3lw76bpj5uz8p6VWJ19gDv+A6GQl7Djsj59QITpudV5iljh3Gg6qNjollAlJxHuHrxlllI 8CeKyOr9mCLjma1RjsGvPG8sbI2AMjcjcf+uROZnyCLFTs0sKIWoHo0GpNfNDE6W 1wIDAQAB -----END PUBLIC KEY-----	1000
-----BEGIN PUBLIC KEY----- MIIBJANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAE624dHsXXyDYGdzWWxFDY s5RNzKxpjSBjhQCTulOir3hEzKs5K5sBuilc8pkjLW/om64RzdHKwOc0e1F9UQP HJWCExHsHY5YHQGfGoghlf3Sez7HHJJNSLSrNoXrTb/aaJbKefAru/1n0b79K0 pWtMfC0iVqihagtgZZclmqfJoulouRdAs7ayPQTLpQOknI2xEu65kzXk3lw76bpj5uz8p6VWJ19gDv+A6GQl7Djsj59QITpudV5iljh3Gg6qNjollAlJxHuHrxlllI 8CeKyOr9mCLjma1RjsGvPG8sbI2AMjcjcf+uROZnyCLFTs0sKIWoHo0GpNfNDE6W 1wIDAQAB -----END PUBLIC KEY-----	1000
-----BEGIN PUBLIC KEY----- MIIBJANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAE624dHsXXyDYGdzWWxFDY s5RNzKxpjSBjhQCTulOir3hEzKs5K5sBuilc8pkjLW/om64RzdHKwOc0e1F9UQP HJWCExHsHY5YHQGfGoghlf3Sez7HHJJNSLSrNoXrTb/aaJbKefAru/1n0b79K0 pWtMfC0iVqihagtgZZclmqfJoulouRdAs7ayPQTLpQOknI2xEu65kzXk3lw76bpj5uz8p6VWJ19gDv+A6GQl7Djsj59QITpudV5iljh3Gg6qNjollAlJxHuHrxlllI 8CeKyOr9mCLjma1RjsGvPG8sbI2AMjcjcf+uROZnyCLFTs0sKIWoHo0GpNfNDE6W 1wIDAQAB -----END PUBLIC KEY-----	1000
-----BEGIN PUBLIC KEY----- MIIBJANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAE624dHsXXyDYGdzWWxFDY s5RNzKxpjSBjhQCTulOir3hEzKs5K5sBuilc8pkjLW/om64RzdHKwOc0e1F9UQP HJWCExHsHY5YHQGfGoghlf3Sez7HHJJNSLSrNoXrTb/aaJbKefAru/1n0b79K0 pWtMfC0iVqihagtgZZclmqfJoulouRdAs7ayPQTLpQOknI2xEu65kzXk3lw76bpj5uz8p6VWJ19gDv+A6GQl7Djsj59QITpudV5iljh3Gg6qNjollAlJxHuHrxlllI 8CeKyOr9mCLjma1RjsGvPG8sbI2AMjcjcf+uROZnyCLFTs0sKIWoHo0GpNfNDE6W 1wIDAQAB -----END PUBLIC KEY-----	30
-----BEGIN PUBLIC KEY----- MIIBJANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAE624dHsXXyDYGdzWWxFDY s5RNzKxpjSBjhQCTulOir3hEzKs5K5sBuilc8pkjLW/om64RzdHKwOc0e1F9UQP HJWCExHsHY5YHQGfGoghlf3Sez7HHJJNSLSrNoXrTb/aaJbKefAru/1n0b79K0 pWtMfC0iVqihagtgZZclmqfJoulouRdAs7ayPQTLpQOknI2xEu65kzXk3lw76bpj5uz8p6VWJ19gDv+A6GQl7Djsj59QITpudV5iljh3Gg6qNjollAlJxHuHrxlllI 8CeKyOr9mCLjma1RjsGvPG8sbI2AMjcjcf+uROZnyCLFTs0sKIWoHo0GpNfNDE6W 1wIDAQAB -----END PUBLIC KEY-----	50

Εικόνα 6: Money Transfers Page

εικόνα 5. Εδώ, μπορεί να δει τα μηνύματα που έχει στείλει (outgoing messages) και με χρήση του κατάλληλου button να δει τα μηνύματα που έχει παραλάβει (ingoing messages). Ομοίως, με την επιλογή Show Money Transfers, οδηγούμαστε στη σελίδα που φαίνεται στην εικόνα 6 όπου κανείς μπορεί να δει εισερχόμενα και εξερχόμενα coin transactions. Τέλος, με χρήση της επιλογής Mining, οδηγούμαστε στην σελίδα της εικόνας 7 από όπου μπορούμε να δούμε και να αλλάξουμε το stake, να δούμε πόσα blocks έχουμε κάνει εμείς mine, πόσα transactions απομένουν για να γεμίσει το επόμενο block και έναν κατάλογο που περιλαμβάνει για τα blocks που έχουμε κάνει mine, την θέση τους στο blockchain και το συνολικό fee που εισπράξαμε από το mining αυτών.

Location in chain	Total Fee
5	89
12	187

Εικόνα 7: Mining Page

Για την χρήση του frontend θα χρειαστεί η εκτέλεση των εντολών `npm install` και `npm start` εντός του φακέλου όπου βρίσκεται το frontend.

3 Μετρήσεις και αποτελέσματα

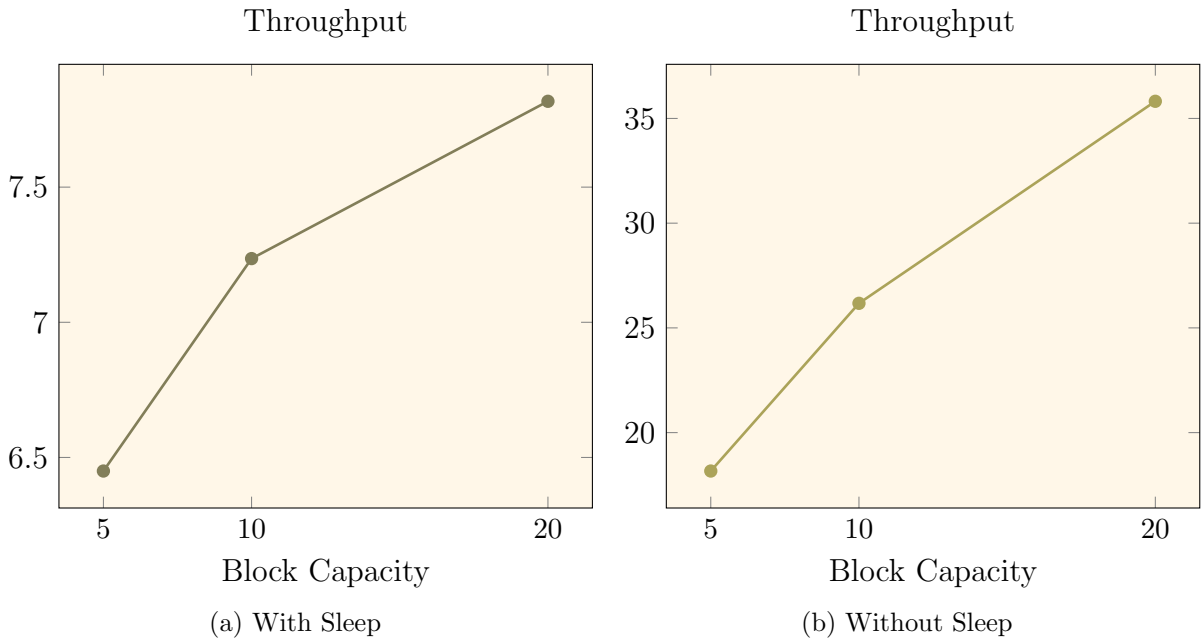
Πραγματοποιούμε μετρήσεις για το σύστημά μας στις 3 περιπτώσεις που υποδεικνύονται. Χρησιμοποιήσαμε 5 VMs από την cloud υπηρεσία okeanos-knossos.

3.1 5 clients - block capacity 5, 10, 20 - ίδια stakes για όλους

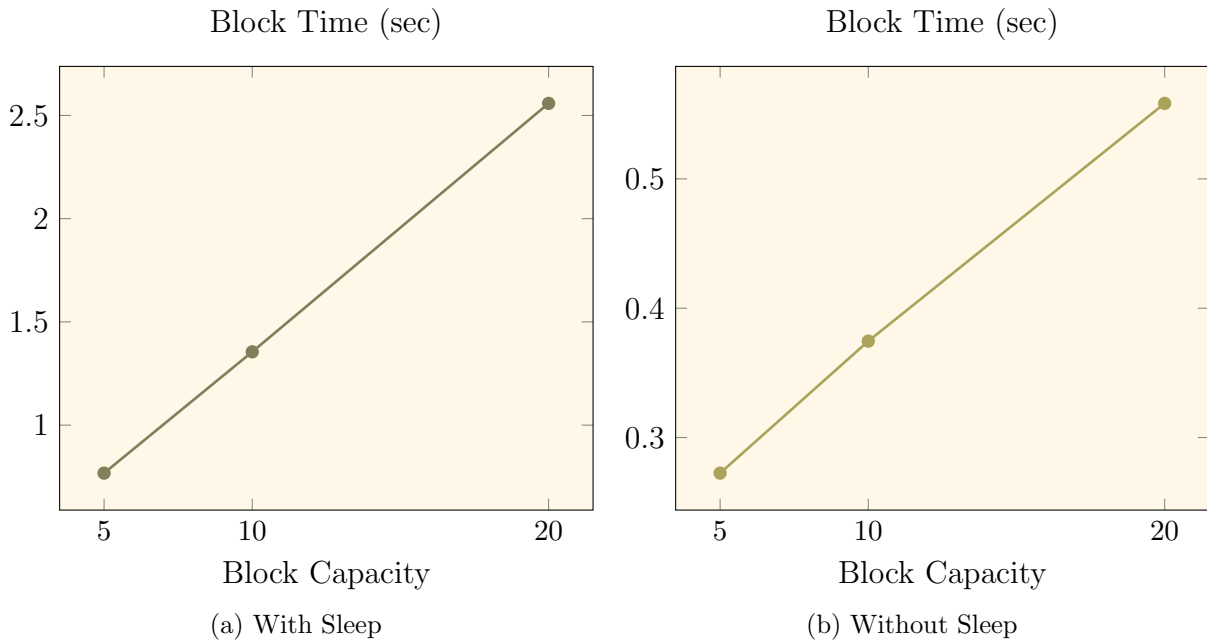
Στα παρακάτω διαγράμματα των εικόνων 8, 9 φαίνεται :

1. Το throughput του συστήματος, δηλαδή το πλήθος transactions που ικανοποιούνται στην μονάδα του χρόνου. Υπολογίστηκε ως το πηλίκο του πλήθους των transactions

(500) δια του συνολικού χρόνου εκτέλεσης (με έναρξη αμέσως μετά τον τερματισμό του bootstrapping).



Εικόνα 8: Throughput με block capacities = {5, 10, 20}, 5 clients και stakes ίσα με 10 για όλους.



Εικόνα 9: Block Time με block capacities = {5, 10, 20}, 5 clients και stakes ίσα με 10 για όλους.

2. To Block Time, δηλαδή τον μέσο χρόνο που μεσολαβεί για την έκδοση νέου block. Υπολογίστηκε ως το πηλίκο του συνολικού χρόνου εκτέλεσης δια του συνολικού πλήθους blocks που δημιουργήθηκαν.

Να σημειωθεί ότι ο proxy (για λόγους αντιμετώπισης deadlocks μεταξύ των κόμβων) εισάγει σκόπιμα καθυστέρηση 0.1 second μεταξύ των requests για νέο transaction. Συνεπώς, ο συνολικό χρόνος εκτέλεσης περιλαμβάνει και αυτόν τον χρόνο. Με αφαίρεση όλων αυτών των 0.1 seconds, τα διαγράμματα που παράγονται είναι αυτά δεξιά (Without Sleep). Αυτό το 0.1 second δεν είναι απαραίτητο στην περίπτωση που δημιουργούνται σειριακά transactions (και όχι από πολλούς ταυτόχρονα, όπως στο συγκεκριμένο test case).

Από τα διαγράμματα, παρατηρούμε ότι με αύξηση του capacity του block αυξάνεται τόσο το throughput όσο και ο μέσος χρόνος δημιουργίας νέου block. Προφανώς, όσο μεγαλύτερο είναι το capacity χρειάζεται περισσότερος χρόνος για να γεμίσει ένα block, ενώ ταυτόχρονα δεν γίνονται και τόσο συχνές δημιουργίες blocks, mining, block broadcasts κλπ. Ως εκ τούτου, περισσότερα transactions μπορούν να εκτελούνται στην μονάδα του χρόνου χωρίς να καθυστερούν εξαιτίας αυτών των διαδικασιών. Ενδεικτικά, επειδή χρησιμοποιήθηκαν τέτοιες πληροφορίες για τους υπολογισμούς μας, παροσιάζεται στην εικόνα 10 το state ενός κόμβου και το συνολικό μέγεθος του blockchain στο τέλος αυτού του πειράματος με capacity 5.

```
MY BLOCKCHAIN : 101
RING AFTER MINING [{ 'publickey': 'b'-----BEGIN PUBLIC KEY-----\nMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA3SZA7bwtrQrGZ8Br2up\nnpNbH0Wrc1EdLddNv7d/OET24LM
BIM74Q1r+ESNGsJ5u41/UFRjT6xG27VNLn\I0DjXpHv9Y/aCgb1b1ONISXb2aGz85/b3vDzn9QDC5G1m16U5nmM2a4CEi5Xe\nnfZ/xHcUNuba5kp7CKiVmDwCdAOW/01QIa/JbKEX0TcM9fHb7sW
xORYn/kTxPQp/n0/y5jyKsVrD/xLMvWTTvjGp9ER9LQ/u4IYHacAJT9WFZj0eyTtLvjTavwvZYP2/nh09a4vHU0LTX/OpPjAz2W2/ms+bwS4vfu0kNaA108xLrkgTfj2sNNip7mh/hzAo\nnVmIDAQAB\n-----END PUBLIC KEY-----', 'id': 0, 'ip': '192.168.2.1', 'port': 5000, 'balance': 1687.0, 'stake': 0}, { 'publickey': 'b'-----BEGIN PUBLIC KEY-----\nMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA8SNjCw3mtAQcnXWLBfsG\nn5yGT+jctJkQeaDHS+HAAH6505nr0yZs1053dKYrJIAA91eg2hxtmho9H+wxw/Qv\nnXG4Pf637Uuy/4EqGHqPMGyCvKX/B4hBg
xQq/f/U3nxsY+dP6mk+UewCb2rQ7k1st/nE9v/tW0aEzplAFyWgh6DF4k8EXh3PsKz\nvnuFxfCpGTwv0f8CevGw/RvgWTS10/nsHrAAHYEE7sKXUDN6NUG7QGa29+8bvqj\nqqA7LSyDfK/7bwKKa7ebk8P
MtxPax89u\nnN+YxwWgFqYx2CQ1bHf8vjYdWw+Igb6WVGFbG/nulBjFMBFa16Ux23vMb3J\nUh44W5nnwIDAQAB\n-----END PUBLIC KEY-----', 'id': 1, 'ip': '192.168.2.2', 'port': 5001, 'balance': 838, 'stake': 0}, { 'publickey': 'b'-----BEGIN PUBLIC KEY-----\nMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAx9YOKk1iah203W5db04D\n\nnxPq0qS2fg/JhBU9E
PaU02uWJ4fH4XCCpuzMTrxqB179B0XZEmEYOP5d182gGgk\nnKfCgF6bH1vzF1DTMy74D0VPkazzKwz7WQGGyYak5T+MthNaIrGkK378dMv+mMxt\nl\nnnSeN34Lm2byQ6Pz/VcakpD+xf1R/QzchW2RxtqQ
puQXMIy1wz25sQ8Q14vu0yYf\nnuy908R80vDXpdEeeZYrLrnnJu2u22DoGLK35QqPRFQZ3wz\nMSLHjz6HcLRcSW050X\nnbdHfUeFVYyzLwKMS1jsA7nzpskM1h1ZXDp\nywDQkDG2tq7oa0yJdm70B0ZHXxKpQ
\nnhwIDAQAB\n-----END PUBLIC KEY-----', 'id': 2, 'ip': '192.168.2.3', 'port': 5002, 'balance': 1047, 'stake': 0}, { 'publickey': 'b'-----BEGIN PUBLIC KEY-----\nMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA3Pm0L0oxB6CKYU2HM7cf\n\nnVLqtpYrYUfKj3uqCHTB2y1ZN6m1uCKAMHgHsn+Sj5UqvUYLSocLJlcjwoBedw66Y\n\nn/nqFV0iDQSL04qZV7PRExdM
qRns4RudsZv2pXWMI43ks11QLO9tZz1Kcy10W5+nd0b\n\nVONhTZXAfaQ15BQZbac/7LDCUhfCPLZcpeVjz2vM23r\n\nrorT20r8NaDLrBqMda\n\nnuX9TwsFYqQNiBKpiaBSSQVF8pjqQ0E5Mh\n\nInMw9S8f9+hm
4zGjPRFRQmbtEHUTK\n\nn51xwCgPRBwdvH9KcaZxlyp88+yzuGTU1aP2kdZNTioj\n\nb11+no0bsqInpctNgrwW\n\nn/IDAQAB\n-----END PUBLIC KEY-----', 'id': 3, 'ip': '192.168.2.4', 'port': 5003, 'balance': 333, 'stake': 0}, { 'publickey': 'b'-----BEGIN PUBLIC KEY-----\nMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAqZdrRvFyaJ5\n\niMc0bVX\n\nniVik+Iio
w7A80E0qo1i\n\nTad+mwK\n\nxw6d10MwK66C7ANxw5\n\nsXc4H\n\naE6A7KqPoT3\n\nnleJp8176YTrpVz6MDMk\n\nx10tccRRng212H3K1\n\nLxVkdIIO8T\n\nuEgF+Rq675QAUU17\n\nn7rRU\n\nh/ZLsew+iwz+Sou18s3cV\n\nXACE
BMR4cpZb7iME\n\nnmpFDEV3Vnc9zEbZ\n\nu42c\n\nng+oAQmZpjuwh0sD\n\nYRS0rTCN5\n\nklzTj\n\nH86wqHMXYS\n\nCp0qI\n\nTanUdoigB/y\n\n+k9Vin\n\nndK/6/zRCg+MDy6W\n\nNEOCttmJq+czAIF6Ggy6U\n\nMTZ95LJGs+L0a\n\nnIzS
ah0danTcc\n\nnj\n\nIDAQAB\n-----END PUBLIC KEY-----', 'id': 4, 'ip': '192.168.2.5', 'port': 5004, 'balance': 1095, 'stake': 0}]
MY BCC 1687.0
```

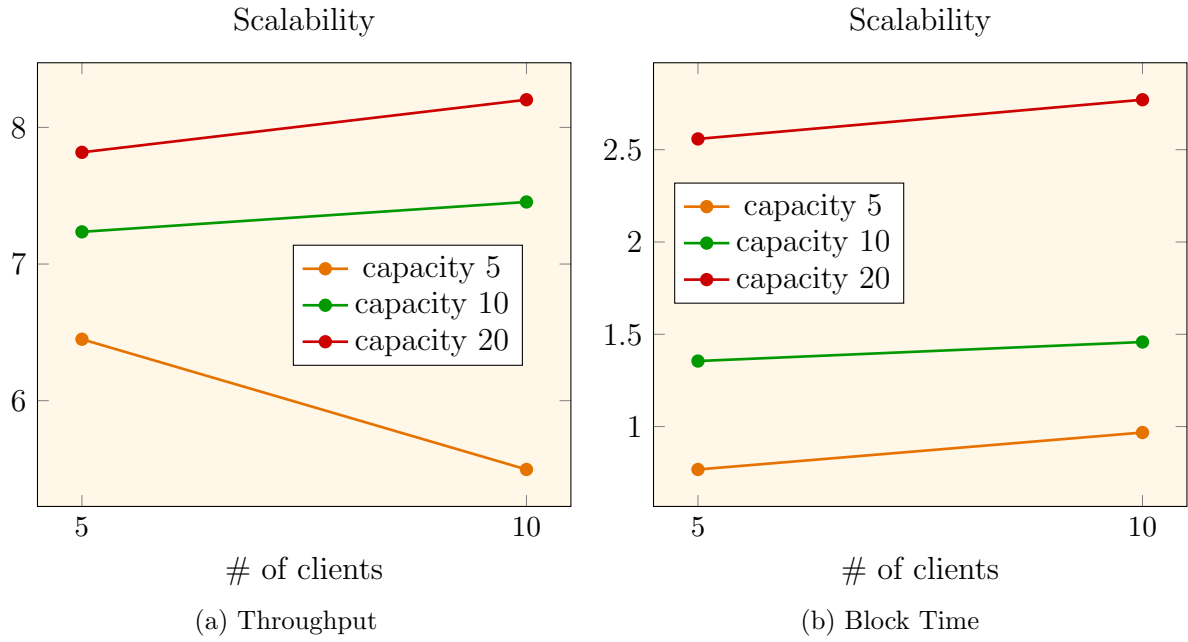
Εικόνα 10: State του κόμβου 0 μετά το πείραμα και μέγεθος του blockchain

3.2 Κλιμακωσιμότητα του Συστήματος

Στη συνέχεια, κάνουμε τα προηγούμενα πειράματα αλλά πλέον με 10 clients. Τα διαγράμματα που παράγονται φαίνονται παρακάτω (εικόνα 11) και απεικονίζουν για κάθε πλήθος clients το throughput και το blocktime για κάθε διαφορετικό block capacity. Παρατηρούμε ότι εν γένει το throughput αυξάνεται λίγο με την αύξηση των clients καθώς υπάρχουν περισσότερα transactions προς εκτέλεση στο σύστημα, αλλά η σειριακή τελικά εκτέλεσή τους δεν επιτρέπει κάποια μεγάλη αύξηση στο throughput. Μάλιστα με capacity 5, παρατηρούμε ότι λόγω των πολύ συχνών minings, το σύστημα καταλήγει να έχει μικρότερο throughput. Ως προς το block time, παρατηρούμε ότι αυτό μένει σχεδόν σταθερό για σταθερό capacity, καθώς λόγω της σειριοποίησης των transactions, δεν υπάρχει λόγος να αλλάξει ο χρόνος που χρειάζεται για να γεμίσει ένα block.

3.3 Δικαιοσύνη

Στη συνέχεια, με χρήση 5 clients και capacity ίσο με 5 (όπως στο πρώτο πείραμα), θέτουμε το stake ενός εκ των participants ίσο με 100 BCC. Στην περίπτωση που όλοι οι nodes είχαν



Εικόνα 11: Σύγκριση αποτελεσμάτων με χρήση 5 και 10 clients

ίδια stakes, μετρούμε πόσες φορές έγινε validator ο καθένας. Το αποτέλεσμα που λαμβάνουμε είναι : [21, 15, 27, 20, 17]. Δηλαδή και οι 5 participants έγιναν περίπου αντιστοιχες φορές validators. Αντιθέτως, αν κάποιος (έστω ο τρίτος κόμβος) έχει stake ίσο με 100, το αποτέλεσμα που λαμβάνουμε είναι : [8, 6, 54, 4, 6], δηλαδή ο τρίτος κόμβος έγινε πολύ περισσότερες φορές validator σχετικά με τους υπόλοιπους. Προφανώς, αυτό είναι άμεσο αποτέλεσμα του consensus αλγορίθμου που χρησιμοποιούμε, δηλαδή το proof of stake, σύμφωνα με το οποίο η πιθανότητα να εκλεγεί κάποιος validator είναι ανάλογη του ποσού stake που δεσμεύει.