Εθνικο Μετσοβιο Πολυτεχνειο
Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων
Τομεας

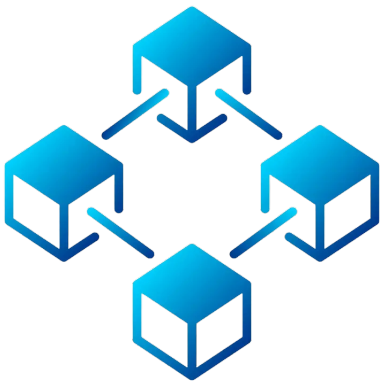
Συγκριτική Απεικόνιση Συστημάτων Blockchain

Solana & Ethereum

Διπλωματικη Εργασια

του

ΚΑΦΙΡΗ Κ. ΑΛΕΞΑΝΔΡΟΥ



Επιβλέπων: Νεκτάριος Κοζύρης
Καθηγητής

Αθήνα, Σεπτέμβριος 2022

Εθνικο Μετσοβιο Πολυτεχνειο
Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων
Τομεας

Συγκριτική Απεικόνιση Συστημάτων Blockchain

Solana & Ethereum

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

ΚΑΦΙΡΗ Κ. ΑΛΕΞΑΝΔΡΟΥ

Επιβλέπων: Νεκτάριος Κοζύρης
Καθηγητής

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 13η Σεπτεμβρίου 2022.

(Υπογραφή)	*(Υπογραφή)*	*(Υπογραφή)*
.....	.....	.....
Νεκτάριος Κοζύρης	Πνευματικάτος Διονύσιος	Τσουμάκος Δημήτριος
Καθηγητής	Καθηγητής	Αναπληρωτής Καθηγητής

Αθήνα, Σεπτέμβριος 2022

Εθνικο Μετσοβιο Πολυτεχνειο
Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων
Τομεας

ΔΗΛΩΣΗ ΜΗ ΛΟΓΟΚΛΟΠΗΣ ΚΑΙ ΑΝΑΛΗΨΗΣ ΠΡΟΣΩΠΙΚΗΣ ΕΥΘΥΝΗΣ

Με πλήρη επίγνωση των συνεπειών του νόμου περί πνευματικών δικαιωμάτων, δηλώνω ενυπογράφως ότι είμαι αποκλειστικός συγγραφέας της παρούσας Πτυχιακής Εργασίας, για την ολοκλήρωση της οποίας κάθε βοήθεια είναι πλήρως αναγνωρισμένη και αναφέρεται λεπτομερώς στην εργασία αυτή. Έχω αναφέρει πλήρως και με σαφείς αναφορές, όλες τις πηγές χρήσης δεδομένων, απόψεων, θέσεων και προτάσεων, ιδεών και λεκτικών αναφορών, είτε κατά κυριολεξία είτε βάσει επιστημονικής παράφρασης. Αναλαμβάνω την προσωπική και ατομική ευθύνη ότι σε περίπτωση αποτυχίας στην υλοποίηση των ανωτέρω δηλωθέντων στοιχείων, είμαι υπόλογος έναντι λογοκλοπής, γεγονός που σημαίνει αποτυχία στην Πτυχιακή μου Εργασία και κατά συνέπεια αποτυχία απόκτησης του Τίτλου Σπουδών, πέραν των λοιπών συνεπειών του νόμου περί πνευματικών δικαιωμάτων. Δηλώνω, συνεπώς, ότι αυτή η Πτυχιακή Εργασία προετοιμάστηκε και ολοκληρώθηκε από εμένα προσωπικά και αποκλειστικά και ότι, αναλαμβάνω πλήρως όλες τις συνέπειες του νόμου στην περίπτωση κατά την οποία αποδειχθεί, διαχρονικά, ότι η εργασία αυτή ή τμήμα της δεν μου ανήκει διότι είναι προϊόν λογοκλοπής άλλης πνευματικής ιδιοκτησίας.

(Υπογραφή)

.....
Καφίρης Αλέξανδρος

13 Σεπτεμβρίου 2022

Περίληψη

Στις μέρες μας γίνεται όλο και συχνότερα λόγος για τα διάφορα συστήματα Blockchain και για τις διάφορες εφαρμογές που προσφέρουν. Αναμφίβολα πρόκειται για έναν κλάδο της Πληροφορικής που προβλέπεται να ανοίξει νέους δρόμους στην αντίληψή μας για τα Κατανεμημένα Συστήματα, αλλά και να δώσει χώρο στην ανάπτυξη νέων κλάδων της Οικονομίας.

Τα τελευταία 10 χρόνια έχουν προκύψει αμέτρητα νέα κρυπτονομίσματα, το κάθε ένα με διαφορετική αρχιτεκτονική και τρόπο λειτουργίας αλλά και διαφορετικό σκοπό. Από το Bitcoin το 2008 μέχρι το Solana το 2017, κάθε ένα ξεχωριστό νέο σύστημα δίνει τροφή για σκέψη και περαιτέρω έρευνα στον τομέα αυτό. Εφαρμογές όπως τα Smart Contracts διευρύνουν τις δυνατότητες των blockchain σε σημείο που σήμερα έχουμε από αμέτρητα συστήματα κρυπτονομισμάτων μέχρι ολόκληρους οικονομικούς οργανισμούς να λειτουργούν εξ' ολοκλήρου πάνω σε τέτοιες πλατφόρμες.

Όλα τα παραπάνω μας οδηγούν αναπόφευκτα να αναζητήσουμε λεπτομέρειες για τον τρόπο λειτουργίας και τις δυνατότητες των ανωτέρω συστημάτων. Σκοπό της εργασίας αυτής θα είναι (α) η αναλυτική μελέτη και παρουσίαση δύο εκ των δημοφιλέστερων blockchains, των Ethereum και Solana και (β) μια συγκριτική μελέτη τους πάνω σε διάφορες παραμέτρους τους όπως αυτές προκύπτουν από την μελέτη τους.

Λέξεις Κλειδιά

Κατανεμημένα Συστήματα, Blockchain, Ethereum, Solana

Abstract

Nowadays, blockchain systems are more and more mentioned when someone want to talk about how technology has evolved. Blockchain systems offer capabilities that were never imagined before. This Computer Science field will undoubtedly pave new roads in our understanding of Distributed Systems and also allow Economics progress as well.

In the last 10 years, numerous new cryptocurrencies have emerged, each of them being unique in terms of architecture and functionality, while also serving a new purpose. Starting from the Bitcoin in 2008 to Solana in 2017, each new system provides as food for thought for further research in that field. Applications enabled with Smart Contracts broaden the capabilities of blockchains resulting in countless existing cryptocurrencies and even complete economic organizations completely based on such blockchain platforms.

As a result, we are very keen to studying further details on the functionality of these systems on a technical level and determine to what extend they can grow. The purpose of this paper is to (a) make a detailed walkthrough and present two of the most famous blockchains currently available, Ethereum and Solana and (b) to compare these two blockchains based on metrics determined from our research.

Keywords

Blockchain, Distributed Systems, Blockchain, Ethereum, Solana

στους γονείς μου

Ευχαριστίες

Θα ήθελα καταρχήν να ευχαριστήσω τον καθηγητή κ. Κοζύρη Νεκτάριο για την επίβλεψη αυτής της διπλωματικής εργασίας και για την ευκαιρία που μου έδωσε να την εκπονήσω στο εργαστήριο Υπολογιστικών Συστημάτων. Επίσης ευχαριστώ ιδιαίτερα την κα. Δόκα Κατερίνα για την καθοδήγησή του και την εξαιρετική συνεργασία που είχαμε. Τέλος θα ήθελα να ευχαριστήσω τους γονείς μου για την καθοδήγηση και την ηθική συμπαράσταση που μου προσέφεραν όλα αυτά τα χρόνια.

Αθήνα, Σεπτέμβριος 2022

Καφίρης Αλέξανδρος

Περιεχόμενα

Κατάλογος Σχημάτων

Κατάλογος Εικόνων

Κατάλογος Πινάκων

Πρόλογος

Η παρούσα εργασία εκπονήθηκε στο εργαστήριο "ΣΛαβ της Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Ηλεκτρονικών Υπολογιστών του Εθνικού Μετσόβιου Πολυτεχνείου. Η παρακάτω δουλειά γίνεται στα πλαίσια μελέτης και κατανόησης των Blockchain συστημάτων και διενεργήθηκε σε άψουγη συνεργασία με τον καθηγητή Κοζύρη Νεκτάριο και την κα. Δόκα Αικατερίνη.

Κεφάλαιο 1

Εισαγωγή

Η Επιστήμη των Υπολογιστών τα τελευταία χρόνια έχει κάνει αλματώδη βήματα προόδου διευρύνοντας συνεχώς το πεδίο της και κατακτώντας νέα μονοπάτια. Η συνεχής έρευνα στο πεδίο έχει φέρει στον κόσμο δυνατότητες που μέχρι πριν λίγα χρόνια μόνο σαν όνειρο θα φάνταζαν. Κλάδοι όπως η Θεωρητική Πληροφορική, η Τεχνολογία Λογισμικού και τα Υπολογιστικά Συστήματα επεκτείνουν συνεχώς τις δυνατότητές τους και εκπλήσσουν με τα αποτελέσματα που φέρνουν στη ζωή μας.

Ένα από τα πιο ενδιαφέροντα πεδία της σύγχρονης Πληροφορικής είναι και τα Κατανεμημένα Συστήματα. Τα Κατανεμημένα Συστήματα αποτελούν σύνολα υπολογιστικών μονάδων οι οποίες συνεργάζονται μεταξύ τους με σκοπό την επίτευξη ενός κοινού στόχου. Τέτοια συστήματα μπορεί να βρίσκονται παντού μπροστά μας στην καθημερινότητα, όπως για παράδειγμα το τηλεφωνικό δίκτυο το οποίο αποτελείται από χιλιάδες συσκευές που συνεχώς ανταλλάσσουν μηνύματα και ικανοποιούν την ανάγκη της επικοινωνίας. Κάποια από αυτά τα συστήματα είναι πολύ απλοϊκά σε μορφή, όπως μια ιστοσελίδα για παράδειγμα, ενώ άλλα κρύβουν μέσα τους τεράστια πολυπλοκότητα και πολύ ιδιαίτερες απαιτήσεις.

1.1 Σχετικά με την εργασία

Η εργασία αυτή αποπειράται να εξετάσει το αντικείμενο του Blockchain που τόσο έχει αναδυθεί την τελευταία δεκαετία και έχει δημιουργήσει τεράστια προοπτική, τόσο στο χώρο της οικονομίας, όσο και στον χώρο της αμιγούς Πληροφορικής. Η επίλυση σύνθετων προβλημάτων με σκοπό την βελτιστοποίηση τέτοιων συστημάτων κάθε άλλο ανεξάρτητη είναι από τους υπόλοιπους κλάδους. Προκειμένου ένα σύστημα blockchain να γίνει αποδεκτό από το ευρύ κοινό θα πρέπει να ικανοποιεί κάποια κριτήρια και η προσπάθεια κατάκτησής τους οδηγεί σε προόδους σε τομείς όπως τα Δίκτυα Υπολογιστών, η Θεωρητική Πληροφορική, η Κρυπτογραφία κα.

Για τους παραπάνω λόγους θεωρήθηκε αξιόλογο να μελετήσουμε δύο διαφορετικά είδη Blockchain τα οποία δημιουργήθηκαν σε διαφορετικούς χρόνους και με διαφορετικό σκοπό και να κάνουμε μια συγκριτική μελέτη πάνω σε αυτά για να δούμε ποιές είναι η τεχνολογίες που ευνοούν το ένα απέναντι στο άλλο. Τα δύο αυτά Blockchain συστήματα είναι το Ethereum και το Solana και είναι άξια προς μελέτη γενικότερα λόγω των προβλημάτων που έλυσαν όταν κυκλοφόρησαν για πρώτη φορά.

Θα μελετήσουμε θεωρητικές πλευρές των παραπάνω συστημάτων και θα υλοποιήσουμε

κάποια μέσα έτσι ώστε να μπορέσουμε να αξιολογήσουμε την απόδοσή τους, να μελετήσουμε τις αδυναμίες τους και να εντοπίσουμε για ποιές εφαρμογές είναι κατάλληλο το καθένα.

1.2 Related Work

Για την πλήρη κατανόηση της θεματολογίας μελετήθηκαν εργασίες που αφορούν τα πρώτα βήματα του blockchain, όπως το κλασσικό paper για το Bitcoin. Επίσης, μελετήθηκαν διάφορες νέες μέθοδοι για την επίτευξη του consensus, της ακεραιότητας του δικτύου και τη διαχείριση των υπολογιστικών πόρων που τα φιλοξενούν. Επίσης, όπου ήταν αναγκαίο μελετήθηκαν άρθα και φορυμς από το διαδίκτυο καθώς αρκετά μεγάλο κομμάτι από τη θεματολογία της εργασίας είναι σύγχρονο και δεν υπάρχει εκτενής καταγραφή του και μελέτη του στη βιβλιογραφία. Το κυριότερο μέρος της εργασίας που προήλθε από το διαδίκτυο αφορά τις τεχνικές λεπτομέρειες για το Solana.

Τέλος, μελετήθηκε σε βάθος η πειραματική βιβλιοθήκη του Blockbench, ενός benchmarking εργαλείου το οποίο δημιουργεί το κατάλληλο περιβάλλον για να εκτελεστούν τα δύο αυτά blockchains και να μπορούν να παρθούν μετρήσεις για τις επιδόσεις τους.

Όλες οι πηγές της εργασίας είναι λεπτομερώς καταγεγραμμένες στην ενότητα της βιβλιογραφίας.

1.3 Διάρθρωση της εργασίας

Η παρούσα εργασία ξεκινάει με μια θεωρητική περιγραφή των Κατανεμημένων Συστημάτων και μια πολύ συνοπτική παρουσίαση του προβλήματος του Blockchain. Επίσης, περιγράφονται βασικές έννοιες οι οποίες θα χρησιμοποιηθούν αρκετά ή θα υπονοούνται στη συνέχεια της εργασίας.

Στο δεύτερο και τρίτο κεφάλαιο θα γίνει μια εκτενής παρουσίαση των δύο διαφορετικών Blockchain που θα εξετάσουμε, στις οποίες ενότητες θα γίνουν ξεκάθαρες οι ιδιαιτερότητες του καθενός συστήματος. Σκοπός των ενοτήτων αυτών θα είναι ο αναγνώστης να έχει μια καθαρή εικόνα της αρχιτεκτονικής της κάθε τεχνολογίας.

Τέλος, στο τέταρτο κεφάλαιο θα παρουσιάσουμε το πρακτικό μέρος της εργασίας το οποίο περιστρέφεται γύρω από την πειραματική αξιολόγηση των δύο συστημάτων σε κομμάτια που αφορούν την επίτευξη συμφωνίας μεταξύ των κόμβων τους, την ταχύτητα εκτέλεσης διάφορων πράξεων καθώς και την εκτέλεση πιο σύνθετων διαδικασιών. Ευελπιστούμε ότι στο τέλος, τα αποτελέσματα θα μπορέσουν να συνδυαστούν με την θεωρητική γνώση που αποκτήσαμε και να βγάλουμε καθαρά συμπεράσματα για τα δύο αυτά συστήματα.

Μέρος Ι

Θεωρητικό Μέρος

Κεφάλαιο 2

Blockchain

Τα Κατανεμημένα Συστήματα ορίζονται σαν ένα σύνολο υπολογιστικών μονάδων οι οποίες συνεργάζονται με σκοπό την επίτευξη μιας συγκεκριμένης λειτουργίας. Μόνο από τον τίτλο μπορεί κανείς να καταλάβει ότι πρόκειται για πολύπλοκα δίκτυα υπολογιστών που το καθένα εκτελεί την ίδια ή διαφορετική λειτουργία και όλα μαζί οφείλουν να επικοινωνούν αποδοτικά. Επομένως, καταλαβαίνουμε ότι δημιουργούνται πολλά ζητήματα για την ομαλή λειτουργία του συνόλου αλλά ταυτοχρόνως τα αποτελέσματα στα οποία μπορεί οδηγηθούμε είναι εντυπωσιακά.

Το κεφάλαιο αυτό αποσκοπεί σε μια συνοπτική παρουσίαση βασικών προβλημάτων στον τομέα αυτό αλλά και μια γρήγορη παρουσίαση του θέματος της εργασίας. Θα κάνουμε επίσης μια επισκόπηση του προβλήματος του Blockchain ενώ θα δούμε και κάποιες από τις πιο γνωστές εφαρμογές του. Τέλος, θα δούμε μερικές γνωστές εφαρμογές των Κατανεμημένων Συστημάτων ώστε να αποκτήσουμε μια καλύτερη εικόνα των προοπτικών που ανοίγονται με την ανάπτυξή τους.

2.1 Βασικά Προβλήματα

Η ομαλή επικοινωνία μεταξύ υπολογιστών δεν έρχεται με μηδενικό κόστος. Προκειμένου να είναι αποδοτική και επωφελής για τους χρήστες θα πρέπει να προσφέρει ταχύτητα, χαμηλό κόστος ενώ ταυτόχρονα να παρέχει την κατάλληλη ασφάλεια από εξωγενείς παράγοντες. Για να επιτευχθούν τα παραπάνω υπάρχουν εμπόδια που πρέπει να ξεπεραστούν, καθώς επίσης και να χρησιμοποιηθούν τα κατάλληλα μέσα για την αντιμετώπισή τους.

2.1.1 Το πρόβλημα του χρόνου

Όπως αναφέρθηκε προηγουμένως, ένα Κατανεμημένο Σύστημα απαρτίζεται από υπολογιστές που συνεργάζονται. Για να έχει πρακτική και ευρεία εφαρμογή ένα τέτοιο σύστημα θα πρέπει να μπορούν να συμμετέχουν σε αυτό υπολογιστές που δεν βρίσκονται απαραίτητα στο ίδιο δίκτυο, στην ίδια περιοχή ή ακόμα και στην ίδια γεωγραφική περιοχή. Θα πρέπει να είναι ένα παγκόσμιο σύστημα.

Ήδη από την παραπάνω περιγραφή ερχόμαστε αντιμέτωποι με ένα πολύ απλό πρόβλημα. Υπολογιστές σε τεράστιες αποστάσεις μεταξύ τους αντιμετωπίζουν το πρόβλημα των καθυστερήσεων του δικτύου και της καταγραφής του χρόνου. Ακόμα και ο πιό αποδοτικός

υπολογιστής μοιάζει άχρηστος όταν το δίκτυο στο οποίο συνδέεται έχει καθυστερήσεις.

Οι καθυστερήσεις που επιφέρει η επικοινωνία μέσω δικτύου αλλά και οι ασυγχρόνιστη ανταλλαγή μηνυμάτων που προκαλείται λόγω διαφορετικών προφιαγραφών στο ηαρδωαρε και στο λογισμικό μπορούν να οδηγήσουν σε μια χαοτική κατάσταση για το δίκτυο. Έχουν γίνει εκτενείς μελέτες για τον τρόπο με τον οποίο ανεξάρτητες οντότητες μπορούν να οργανώσουν τα ποικίλα μηνύματα που ανταλλάσσουν ώστε να βγάζει νόημα η μεταξύ τους επικοινωνία.

Στην πράξη υπάρχουν διάφοροι τρόποι που μπορεί να επιτευχθεί ο συγχρονισμός. Ο πιο απλός είναι η χρήση ενός κοινού σημείου αναφοράς όπως η χρήση της παγκόσμιας ώρας UTC. Παρόλα αυτά αυτό δεν είναι πάντα μια αποδεκτή λύση καθώς ο κάθε υπολογιστής μπορεί να έχει σφάλματα στον υπολογισμό του χρόνου και εν τέλει να οδηγηθούμε και πάλι σε αποσυγχρονισμό. Άλλοι τρόποι είναι η χρήση πιο πολύπλοκων δομών όπως τα λογικά ρολόγια ή τα ρολόγια Lamport. Ωστόσο, τέτοιες μέθοδοι εισάγουν πολυπλοκότητα στο σύστημα.

Από τα παραπάνω βλέπουμε ότι δεν υπάρχει κάποια πεπατημένη οδός για να καταλήξουμε σε μια σίγουρη και σταθερή λύση. Άλλες μέθοδοι όπως οι εκλογές μεταξύ των κόμβων για τον ορισμό κάποιου σαν ρυθμιστή της επικοινωνίας ή η εφαρμογή αμοιβαίου αποκλεισμού μεταξύ κόμβων όταν πάνε να εκτελέσουν εργασίες πάνω στα ίδια δεδομένα μπορούν να εφαρμοστούν, ωστόσο το μόνο σίγουρο είναι ότι η επιλογή που θα πρέπει να γίνει εξαρτάται άμεσα από τις σχεδιαστικές μας επιλογές και την πολυπλοκότητα που θέλουμε να δώσουμε στο σύστημά μας.

2.1.2 Ανάκαμψη από σφάλματα

Όταν συνυπάρχουν πολλοί υπολογιστές σε ένα δίκτυο είναι αναμενόμενο ότι κατά διαστήματα κάποιοι θα έχουν σφάλματα και δεν θα μπορούν να συμμετάσχουν στις λειτουργίες του συστήματος. Τί συμβαίνει όμως όταν κάποιος υπλογιστής αποσύρεται ενώ οι υπόλοιποι δεν έχουν λάβει κάποια ειδοποίηση ή ακόμα όταν περιμένουν κάποια σημαντική πληροφορία από αυτόν· Τί συμβαίνει όταν περισσότεροι από έναν κόμβο βγουν ταυτόχρονα ή πολύ γρήγορα από το δίκτυο·

Τα παραπάνω ερωτήματα δεν έχουν συγκεκριμένη απάντηση. Η αντιμετώπιση μπορεί να κυμαίνεται από την άμεση εύρεση αναπληρωματικών κόμβων μέχρι την κύρηξη αδυναμίας απόκρισης του συστήματος. Στην συνέχεια της εργασίας θα δούμε πώς τα συστήματα blockchain μπορεί να αντιμετωπίσουν τέτοια ζητήματα.

2.1.3 Consensus

Στην Πληροφορική, με τον όρο consensus αναφερόμαστε στην επίτευξη συμφωνίας μεταξύ κόμβων ενός δικτύου τη στιγμή που ένα μέρος των κόμβων αυτών μπορεί να είναι κακόβουλος. Η συμφωνία μπορεί να αφορά οτιδήποτε, από μια απλή επικύρωση κάποιας πληροφορίας μέχρι την απόφαση για κάποια ενέργεια που θα πρέπει να πάρει συλλογικά το δίκτυο. Όπως σε όλα τα προβλήματα, η λύση κι εδώ δεν είναι μοναδική.

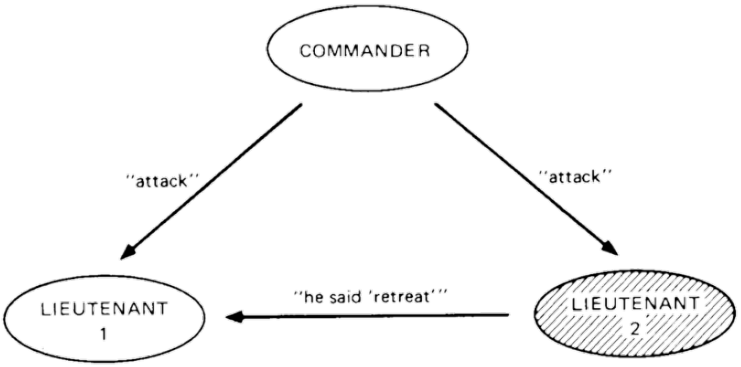
Το πρόβλημα των Βυζαντινών Στρατηγών

Το πρόβλημα των Βυζαντινών Στατηγών διατυπώθηκε από τον L.Lamport [1] και συνοψίζει την παραπάνω περιγραφή του προβλήματος του ϛονσενσυς σε ένα πολύ πρακτικό και καταβοητό σχήμα. Υποθέτουμε ότι ένας στρατός χωρισμένος σε n φάλαγγες έχει κατασκηνώσει απέναντι από ένα αντίπαλο στρατόπεδο. Η κάθε φάλαγγα έχει έναν υπολοχαγό που την οδηγεί και ολόκληρος ο στρατός έχει έναν διοικητή που αποφασίζει για το αν θα επιτεθεί ή όχι στον εχθρό.

Ο σκοπός είναι να παρθεί μια απόφαση για το αν θα γίνει η επίθεση ή όχι. Στο πρόβλημα υπάρχουν δύο βασικές παράμετροι. Πρώτον, οι υπολοχαγοί και οι διοικητές επικοινωνούν μόνο μέσω μηνυμάτων μεταξύ τους τα οποία μεταφέρονται μέσω αγγελιαφόρων. Δεύτερον, υποθέτουμε ότι οποιοφήποτε διοικητικό μέρος μπορεί να είναι διεφθαρμένο και να θέλει να υποκινήσει διαφορετική κίνηση από τον δικοικητή. Σκοπός του προβλήματος είναι να βρεθεί ένας αλγόριθμος επικοινωνίας έτσι ώστε:

- Όλοι οι πιστοί διοικητικοί να αποφασίσουν για το ίδιο σχέδιο δράσης.
- Ένας μικρός αριθμός προδοτών να μην μπορεί να επηρεάσει την συνολική απόφαση.

Για το παραπάνω πρόβλημα ο Lamport απέδειξε ότι όταν υπάρχουν 3 μέρη και το ένα τουλάχιστον είναι κακόβουλο, τότε δεν μπορεί να επιτευχθεί συμφωνία για κοινό σχέδιο δράσης.

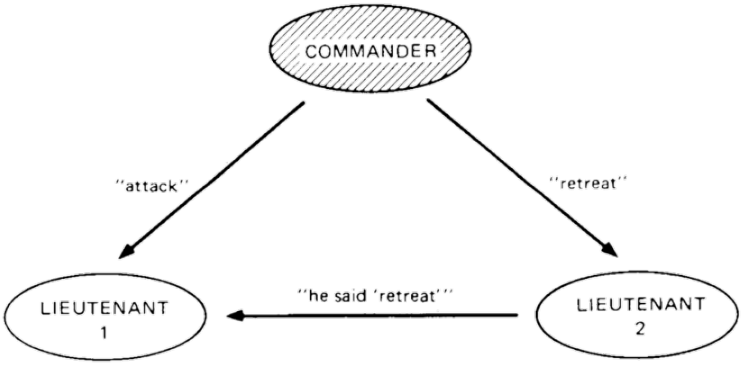


Σχήμα 2.1: *Ο ένας υπολοχαγός είναι προδότης*

Εδώ βλέπουμε ότι ενώ ο διοικητής στέλνει την ίδια εντολή στους 2 λοχαγούς, ο ένας στέλνει το αντίθετο στον άλλον λοχαγό και τελικά δεν είναι εφικτό να παρθεί μια απόφαση.

Και στην περίπτωση που ο διοικητής είναι προδότης και στέλνει διαφορετικό μήνυμα στους δύο υπολοχαγούς βλέπουμε ότι δεν μπορεί να έρθει συμφωνία γιατί το κάθε μέρος λαμβάνει διαφορετικές εντολές.

Ωστόσο, παρά την αδυναμία επίλυσης του προβλήματος στην περίπτωση ενός προδότη για όταν υπάρχουν 3 μέρη, ο Lamport επίσης απέδειξε ότι το πρόβλημα δέχεται λύση και οι κόμβοι μπορούν να καταλήξουν σε συμφωνία όταν υπάρχουν συνολικά $3m + 1$ κόμβοι και από αυτούς το πολύ m είναι οι προδότες.



Σχήμα 2.2: *Ο διοικητής είναι προδότης*

Byzantine Fault Tolerance

Ένα σύστημα λέμε ότι είναι Byzantine Fault Tolerant, δηλαδή ανεκτικό σε Βυζαντινά σφάλματα, όταν μπορεί να έρθει σε consensus παρά την ύπαρξη κακόβουλων κόμβων. Είναι πολύ σημαντικό τα συστήματα που κατασκευάζουμε να έχουν αυτή την ιδιότητα ειδικά στην περίπτωση που συμμετέχουν μη αναγνωρισμένοι κόμβοι. Χαρακτηριστικό παράδειγμα είναι τα δημόσια συστήματα blockchain που θα μελετήσουμε στην εργασία. Εκεί οι κόμβοι μπορούν να μπαίνουν ελεύθερα στο σύστημα χωρίς την επιβεβαίωση της ταυτότητάς τους και άρα οι προθέσεις τους να είναι αδιευκρίνιστες.

Βλέπουμε λοιπόν ότι ένα Κατανεμημένο Σύστημα έχει πολλές πλευρές που πρέπει να λάβουμε υπόψιν μας κατά τον σχεδιασμό. Επιπλέον, το είδος του συστήματος και οι απαιτήσεις που έχουμε από αυτό είναι καθοριστικά για τις επιλογές που θα κάνουμε για την επίλυση του κάθε επιμέρους περοβλήματος.

2.2 Blockchain

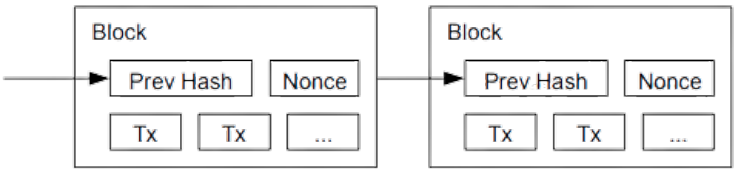
Ο κόσμος της Πληροφορικής ήρθε πρώτη φορά σε επαφή με τον κόσμο του blockchain το 2008 μέσα από το άρθρο του Satoshi Nakamoto σχετικά την αναγκαιότητα δημιουργίας ενός κρυπτονομίσματος. Σκοπός του άρθρου ήταν να τονίσει την αναγκαιότητα αλλά και τη δυνατότητα της ύπαρξης μια ανεξάρτητης οικονομικής οντότητας η οποία θα επιτρέπει στα μέρη της να εκτελούν συναλλαγματικές διαδικασίες. Το άρθρο παρουσίασε μια δομή για ένα τέτοιο σύστημα το οποίο ονομαζόταν Bitcoin [2].

2.2.1 Τί είναι το blockchain·

Το blockchain αποτελεί μια λίστα από εγγραφές, οι οποίες ονομάζονται blocks, οι οποίες συνδέονται μεταξύ τους μέσω κρυπτογραφίας. Αυτή η λίστα θα πρέπει να είναι πολύ δύσκολο έως αδύνατο να αλλοιωθεί πέρα από το τέλος της, στο οποίο συνεχώς εισάγονται νέα blocks.

Το κάθε block μπορεί να περιέχει οποιουδήποτε είδους δεδομένα ανάλογα με το σκοπό που εξυπηρετεί το σύστημα. Το βλοσκσηαιν είναι μια κατανεμημένη εφαρμογή διότι σε αυτό συμμετέχουν πολλοί υπολογιστές οι οποίοι συνεισφέρουν στη συνέχιση της αλυσίδας,

συνήθως με κάποιο σκοπό κέρδους. Η πιο ευρεία χρήση του αφορά την οικονομική σκοπιά και κυρίως τα κρυπτονομίσματα. Ωστόσο, τα τελευταία χρόνια συνεχώς δημιουργούνται νέοι τύποι blockchain οι οποίοι έχουν και διαφορετική στόχευση.



Εικόνα 2.1: *Τα blocks συνδέονται μεταξύ τους*

Το blockchain λειτουργεί σαν ένα δίκτυο ομότιμων κόμβων που σημαίνει ότι όλοι συμμετέχοντες σε αυτό έχουν την ίδια θέση στην ιεραρχία και ο καθένας μπορεί να συνεισφέρει στο δίκτυο, είτε στην κατασκευή της αλυσίδας είτε στην επικύρωση στην συμμετοχή στο consensus.

Δημόσια και Ιδιωτικά blockchains

Δημόσια είναι τα blockchains στα οποία έχει πρόσβαση και συμμετοχή οποιοσδήποτε το επιθυμεί, ενώ τα δεδομένα τους είναι διαφανή, δηλαδή μπορεί ο καθένας να τα δει. Τέτοια είναι τα γνωστά Bitcoin, Ethereum, Solana, Cardano κα.

Αντιθέτως, υπάρχουν και τα ιδιωτικά blockchains στα οποία όπως προκύπτει από την ονομασία η πρόσβαση είναι ελεγχόμενη και οι χρήστες υπόκεινται σε κάποιου είδους αυθεντικοποίηση.

Όπως έχει αναφερθεί και στην προηγούμενη ενότητα, τα δημόσια blockchains είναι πιό επιρεπτή σε βυζαντινά σφάλματα καθώς η έλλειψη ελέγχου των χρηστών δίνει το ελεύθερο σε κακόβουλους κόμβους να συμμετέχουν και να εκμεταλλευτούν το δίκτυο. Ο πιο χαρακτηριστικός τρόπος επίθεσης στο δίκτυο είναι το double-spending.

Το double-spending πρόβλημα περιγράφεται σαν μια επίθεση κατά την οποία ένας κόμβος εκτελεί μια συναλλαγή A και στη συνέχεια μια συναλλαγή B και προσπαθεί να πείσει το δίκτυο ότι μόνο η B υπήρξε και εκτελέστηκε. Αναλόγως το είδος του blockchain τέτοιου είδους επιθέσεις αντιμετωπίζονται με διαφορετικό τρόπο, πάντα όμως με γνώμονα τη διατήρηση της σωστής σειράς στις συναλλαγές.

2.2.2 Bitcoin

Το Bitcoin ήταν το πρώτο blockchain που χρησιμοποιήθηκε ευρεύως με στόχευση την επίλυση ενός οικονομικού προβλήματος. Από το 2008 που έγινε προσβάσιμο έχει αποτελέσει σημείο αναφοράς για τον κλάδο και παρόλη τη συνεχή ανακοίνωση νέων ειδών blockchain, που φαινομενικά έχουν καλύτερες επιδόσεις, παραμένει κυρίαρχο στην αντίληψη που έχει το κοινό για την τεχνολογία αυτή, ενώ ταυτόχρονα αποτελεί και σημείο αναφοράς για την έρευνα πάνω στον τομέα.

Μέσα από το Bitcoin έγινε κατορθωτό ανώνυμοι χρήστες να μπορούν αν ανταλλάσσουν χρηματικές αξίες μεταξύ τους χωρίς να μεσολαβεί κάποιος ιδρυματικός οργανισμός όπως οι

τράπεζες. Ταυτόχρονα, έδωσε τη δυνατότητα στους χρήστες να έχουν πλήρη εποπτεία του δικτύου και να μπορούν να ακολουθούν την πορεία των συναλλαγών τους.

Ως δημόσιο δίκτυο, έπρεπε ωστόσο να λύσει το πρόβλημα του consensus. Για το σκοπό αυτό εφάρμοσε το Proof of Work, ένα κρυπτογραφικό αλγόριθμο ο οποίος εξασφαλίζει ότι για να εισάγει κάποιος χρήστης ένα νέο block στην αλυσίδα, τότε θα πρέπει να έχει αφιερώσει υπολογιστικούς πόρους. Επιπλέον, για την αποφυγή κακόβουλων συμπεριφορών έδωσε χρηματικά κίνητρα στους κόμβους ώστε να συμμετέχουν στη διαδικασία δημιουργίας νέων block. Έτσι δημιουργήθηκε ο όρος του miner, ο οποίος επί της ουσίας αποτελούσε έναν κόμβο που μέσω Proof of Work προσπαθούσε να λύσει ένα κρυπτογραφικό παζλ με σκοπό την εύρεση μιας τιμής η οποία θα μπορούσε να ενώσει το βλοσκ που θέλει να κατασκευάσει με το τέλος της αλυσίδας.

Παρά την τεράστια καινοτομία που εισήγαγε, το Bitcoin έχει δεχτεί τα τελευταία χρόνια έντονη κριτική καθώς το μεγάλο πλήθος κόμβων που διαθέτει έχει οδηγήσει το mining να είναι μια πολύ κοστοβόρα διαδικασία και ταυτόχρονα πολύ επιβαρυντική για το περιβάλλον. Υπολογίζεται ότι οι κόμβοι που συμμετέχουν στο μινινγ χρησιμοποιούν τόση ηλεκτρική ενέργεια που θα μπορούσε να τροφοδοτήσει μια μικρή χώρα. Επιπλέον, είναι ένα αργό δίκτυο και οι δυνατότητες του περιορίζονται σε απλές συναλλαγές μεταξύ χρηστών.

2.2.3 Ethereum

Το Ethereum[3] δημιουργήθηκε το 2013 και η αφορμή για τη δημιουργία του ήταν η αδυναμία του Bitcoin να εκτελέσει κάτι παραπάνω από απλές χρηματικές συναλλαγές. Έτσι, το Ethereum πρότεινε τα έξυπνα συμβόλαια ή αλλιώς smart contracts, τα οποία αποτελούν προγράμματα τα οποία εκτελούνται πάνω στο σύστημα.

Σε αντίθεση με το Bitcoin, η κατάσταση του συστήματος δεν περιέχει πλέον μόνο συναλλαγές αλλά και λογαριασμούς οι οποίοι μπορεί να έχουν σκοπό είτε τις χρηματικές συναλλαγές, είτε την εναπόθεση συμβολαίων. Με τη χρήση των smart contracts έγινε πλέον η δυνατότητα ανάπτυξης εφαρμογών που εκτελούνται αποκεντρωμένα και όλοι μπορούν να συμμετέχουν σε αυτές. Γνωστές τέτοιες εφαρμογές είναι τα token systems όπου αποτελούν νέες χρηματικές μονάδες που "πατάνε" πάνω στο Ethereum και μπορούν να αντικαταστήσουν το επίσημο νόμισμά του, τα DAOs όπου αποτελούν οικονομικούς οργανισμούς που μπορούν να διανέμουν χρήματα στους συμμετέχοντές τους, να κάνουν εκλογές και να λαμβάνουν συλλογικά αποφάσεις, καθώς και η κατανεμημένη αποθήκευση αρχείων.

Το Ethereum συνεχίζει να χρησιμοποιεί δομή παρόμοια με το Bitcoin, δηλαδή εξακολουθεί να χρησιμοποιεί Proof of Work για την επίτευξη consensus, ωστόσο έχει εισάγει αρκετή πολυπλοκότητα καθώς υποστηρίζει την εκτέλεση των smart contracts. Επίσης έχει καλύτερες επιδόσεις από το Bitcoin από άποψη χρόνου εκτέλεσης συναλλαγών, ενώ αυτή την περίοδο οδεύει προς την αλλαγή του πρωτοκόλλου consensus σε Proof of Stake στοχεύοντας σε ακόμα καλύτερες επιδόσεις.

Παρά τις καινοτομίες του όμως, η χρήση Proof of Work οδηγεί σε πολύ μεγάλες προμήθειες συναλλαγών, ή φεες, οι οποίες το κάνουν απωθητικό για κάποιον που δεν διαθέτει το αντίστοιχο κεφάλαιο. Επιπλέον, η εκτέλεση ακόμα και του πιο απλού smart contract μπορεί να έχει κόστος τάξεις μεγέθους μεγαλύτερο από το να έτρεχε μια τέτοια εφαρμογή σε

ένα cloud περιβάλλον.

2.2.4 Solana

Το Solana είναι ένα νέου είδους blockchain το οποίο είναι σχεδιασμένο πάνω σε τελείως διαφορετικές αρχές από ότι τα προγενέστερά του blockchain. Υποστηρίζει τις ίδιες λειτουργικότητες με τα προϋπάρχοντα blockchain, ωστόσο η αρχιτεκτονική του επιτρέπει πολύ μεγάλες ταχύτητες συναλλαγών. Ενδεικτικά, ο χρόνος επικύρωσης μιας συναλλαγής στο Solana είναι περίπου 400μς, ενώ στο Ετηερευμ 15s και στο Bitcoin 10 λεπτά.

Ο τρόπος με τον οποίο κατάφερε να αυξήσει τόσο πολύ τις επιδόσεις ενός blockchain είναι ο συνδυασμός των πρωτοκόλλων Proof of History και Proof of Stake για να επιτευχθεί το consensus. Εδώ ο όρος του mining πάυει να ισχύει και η αλυσίδα μεγαλώνει μέσα από τη συμφωνία μεταξύ των κόμβων του δικτύου και την ύπαρξη ενός αντικειμενικού μέτρου του χρόνου μέσα από το Proof of History. Τα νέα αυτά πρωτόκολλα δεν απαιτούν τη χρήση πολών υπολογιστικών πόρων, ειδικά να κανείς τους συγκρίνει με τα άλλα blockchains, επομένως τα τέλη για την εκτέλεση των συναλλαγών και των προγραμμάτων να είναι πολύ χαμηλά. Ενδεικτικά, τα τέλη για μια μεταφορά χρημάτων μεταξύ δύο λογαριασμών ανέρχονται αυτή τη στιγμή στα 0.00025$.

Οι τόσο καλές επιδόσεις όμως έχουν αντίκτυπο στην αρχιτεκτονική του συστήματος η οποία είναι πολύ σύνθετη και η καμπύλη εκμάθησης της τεχνολογίας είναι πολύ πιο απότομή σε σχέση με τα προαναφερθέντα blockchains. Επίσης, ακόμα και για την δημιουργία ενός απλού προγράμματος απαιτούνται αρκετές γραμμές κώδικα και χαμηλού επιπέδου πράξεις.

Τέλος, οι τόσες νέες τεχνολογίες έχουν επιφέρει μια αστάθεια στο σύστημα το οποίο τον τελευταίο καιρό έχει παγώσει δύο φορές σε σύντομο χρονικό διάστημα.

2.3 Ζητήματα στο blockchain

Η αυξανόμενη δημοφιλία των blockchain συστημάτων, πέρα από τις μεγάλες βελτιώσεις που φέρνει στο χώρο της τεχνολογίας, έχει δημιουργήσει πολλά ερωτήματα για διάφορα θέματα.

Έλεγχος συναλλαγών

Η έλλειψη μιας κεντρικής αρχής, η ανωνυμία των χρηστών και η ελευθερία των συναλλαγών δημιουργούν νομικά θέματα τα οποία αυτή τη στιγμή δεν μπορούν να ελεγχθούν από κάποιο πλαίσιο νόμου. Ο συνδυασμός του ότι η τεχνολογία είναι πολύ σύγχρονη και η μεταβλητότητά της δεν επιτρέπουν τη δημιουργία ενός συγκεκριμένου πλαισίου που θα ρυθμίζει τον τρόπο λειτουργίας του.

Επιπλέον, τα διάφορα συμβόλαια που μεταβάλλουν σταδιακά το ρόλο του blockchain το καθιστούν ακόμα πιο θολό όσον αφορά τη διαχείρισή του. Χαρακτηριστικό παράδειγμα είναι η δημιουργία της δυνατότητας όχι μόνο να αναλλάσσεται το βασικό νόμισμα του κάθε blockchain αλλά να μπορούν να ανταλλάσονται και άλλα νομίσματα, τα tokens. Στην πλειοψηφία τους, αυτά δεν ελέγχονται από κάποιον οργανισμό και ανταλλάσσονται μέσα από αποκεντρωμένα ανταλλακτήρια που λειτουργούν μέσω contracts.

Επεκτασιμότητα

Λόγω της λειτουργίας τους σαν δίκτυα ομότιμων κόμβων, τα περισσότερα blockchains απαιτούν την ανταλλαγή πολλών μηνυμάτων μεταξύ των κόμβων τους με αποτέλεσμα πολλές φορές η επικοινωνία αυτή να γίνεται bottleneck για το σύστημα. Από ένα σημείο και μετά, η εισαγωγή νέων κόμβων στο σύστημα μπορεί να οδηγήσει σε μεγάλες καθυστερήσεις στο δίκτυο καθιστόντας το μη φιλικό προς τους χρήστες.

Δυνατότητα αντικατάστασης οικονομικών οργανισμών

Η συσσώρευση κεφαλαίου στα διάφορα blockchains δημιουργεί το ερώτημα αν αυτά θα μπορέσουν να αποκτήσουν μια πιό απτή υπόσταση και να χρησιμοποιηθούν ευρέως στην καθημερινότητα από τους χρήστες στην πραγματική τους ζωή. Η πραγματικότητα είναι ότι λίγα είναι αυτά που προσφέρουν ταχύτητα συναλλαγών όμοια με αυτή των κεντρικών υπηρεσιών όπως η Visa και η Mastercard.

Ασφάλεια

Η διαφάνεια που υπάρχει στα blockchain συστήματα μπορεί να οδηγήσει σε δύσκολες καταστάσεις κατόχους μεγάλων ποσών χρημάτων. Αν κάποιος μπορεί να συμπεράνει την ταυτότητα ενός χρήστη, τότε θα μπορεί να γνωρίζει σε ποιόν αντιστοιχεί το συγκεκριμένο ποσό.

Επιπλέον, η κατοχή των αναγνωριστικών ενός χρήστη σημαίνει συνολικό έλεγχο σε όλα τα χρηματικά του αποθέματα. Ωστόσο, η πραγματικότητα είναι πως το ίδιο πρόβλημα ισχύει και με τις υπάρχουσες κεντρικές οικονομικές δομές.

Κεντρικός έλεγχος του δικτύου

Σε ένα δίκτυο blockchain ο καθένας μπορεί να συμμετάσχει στην παραγωγή νέων blocks για την αλυσίδα. Επίσης, είναι γνωστός ο προβληματισμός ότι αν κάποιος κατέχει πάνω από το 50% της υπολογιστικής δύναμης, τότε θα μπορεί να ελέγχει πλήρως τις συναλλαγές. Έτσι, υπάρχει ο κίνδυνος να περάσουμε σε κεντρικά διαχειριζόμενα blockchains κάτι το οποίο αναιρεί το σκοπό της ύπαρξής τους. Τα παραπάνω ισχύουν κυρίως σε δίκτυα που χρησιμοποιούν Proof of Work.

Κεφάλαιο 3

Ethereum

3.1 Εισαγωγή

Το Ethereum αποτέλεσε το πρώτο blockchain σύστημα που χρησιμοποιήθηκε για να επεκτείνει τις δυνατότητες των τότε χρησιμοποιούμενων blockchains με την επιτυχημένη χρήστη των smart contracts. Σύμφωνα με την τεχνική περιγραφή του [4] Ethereum, ο κύριος σκοπός που αυτό εξυπηρετεί είναι το να αποτελέσει ένα εικονικό υπολογιστή του οποίου το state ανά πάσα στιγμή είναι το state του blockchain.

Με τη χρήση των smart contracts, το blockchain απέκτησε μια νέα μορφή στην οποία οι χρήστες δεν περιορίζονται μόνο στη διεκπεραίωση συναλλαγών, όπως στο Bitcoin, αλλά μπορούν να δημιουργήσουν τη δική τους λογική εκφρασμένη σε κώδικα και να εκτελέσουν πιο σύνθετες διαδικασίες. Σε συνδυασμό με την ακεραιότητα που προσφέρει ένα blockchain σύστημα, το Ethereum μπόρεσε να ανοίξει το δρόμο για εφαρμογές όπως τα Token Systems, τα DAOs ή ακόμα και την Κατανεμημένη Αποθήκευση Αρχείων.

Ο μηχανισμός που χρησιμοποιείται για τη διασφάλιση της ακεραιότητας του δικτύου παραμένει το Proof Of Work, όμως η επικύρωση των συναλλαγών γίνεται πιο περίπλοκη καθώς πλέον οι miners πρέπει πέρα από τις κλασσικές μεταφορές αξίας να ελέγξουν και να εκτελέσουν τις συναλλαγές που στοχοποιούν τα smart contracts καθώς επίσης και να καταγράψουν τις επιπτώσεις τους στην κατάσταση του συστήματος.

Στο κεφάλαιο αυτό θα ασχοληθούμε και θα αναλύσουμε τον τρόπου με τον οποίο το Ethereum συντηρεί το state του συστήματος, τον τρόπο με τον οποίο επιτυγχάνεται το consensus μεταξύ των κόμβων, ενώ θα δούμε και πώς το σύστημα διαχειρίζεται τις συναλλαγές και αλληλεπιδρά με τους χρήστες. Όλα τα παρακάτω θα μας βοηθήσουν να αποκτήσουμε μια βαθύτερη κατανόηση του συστήματος έτσι ώστε να μπορέσουμε να επικεντρωθούμε σε εκείνα τα σημεία που θα μας δώσουν τη δυνατότητα να αξιολογήσουμε τις επιδόσεις του.

3.2 Τεχνικά Χαρακτηριστικά

Σε αυτό το μέρος θα ασχοληθούμε με τα πιο τεχνικά χαρακτηριστικά του Ethereum. Θα εξετάσουμε τα είδη των κόμβων του, τον τρόπο που αποθηκεύουν το state, την εκτέλεση του consensus μεταξύ των κόμβων καθώς επίσης και τον τρόπο διεκπεραίωσης των συναλλαγών.

3.3 Ο σκοπός του Ethereum

Όπως αναφέρθηκε, το Ethereum στοχεύει στο να αποτελέσει έναν εικονικό υπολογιστικό υπολογιστή, ο οποίος θα αλληλεπιδρά με τους εξωτερικούς χρήστες και θα διαχειρίζεται την κατάσταση του συστήματος ανάλογα με τις απαιτήσεις που υπάρχουν. Για το λόγος αυτό, χρησιμοποιεί ένα νόμισμα, το Ether, το οποίο πέρα από την αντικειμενικη του χρησιμότητα ως χρηματική αξία, αποτελεί και τον τρόπο που οι χρήστες μπορούν να πληρώσουν ώστε να δεσμεύσουν και να χρησιμοποιήσουν του πόρους του συστήματος. Η μικρότερη χρηματική αξία στο σύστημα είναι το Wei και η μεγαλύτερη είναι το Ether. Υπάρχουν και άλλες ενδιάμεσες μονάδες που παρουσιάζονται παρακάτω:

Μονάδα	Ether	Wei
Ether	1	1,000,000,000,000,000,000
Finney	0,001	1,000,000,000,000,000
Szabo	0,000,001	1,000,000,000,000
Shannon	0,000,000,001	1,000,000,000
Lovelace	0,000,000,000,001	1,000,000
Babbage	0,000,000,000,000,001	1,000
Wei	0,000,000,000,000,000,001	1

Πίνακας 3.1: *Χρηματικές μονάδες στο Ethereum*

3.4 Διαχείριση της κατάστασης του συστήματος

Τη στιγμή της συγγραφής της εργασίας, έχουν πραγματοποιηθεί περισσότερες από 1,5 εκατομμύρια συναλλαγές στο δίκτυο από τη στιγμή που διατέθηκε προς χρήση, επομένως είναι εύλογη η απορία του πώς μπορεί να διατηρηθούν όλες αυτές οι συναλλαγές και γίνονται οι απαραίτητοι έλεγχοι ώστε να διασφαλιστεί η εγκυρότητα των νέων συναλλαγών. Για το λόγο αυτό, το Ethereum προχώρησε στον συνδυασμό δομών δεδομένων για να πετύχει το σκοπό αυτό. Οι δομές αυτές αναλύονται στη συνέχεια.

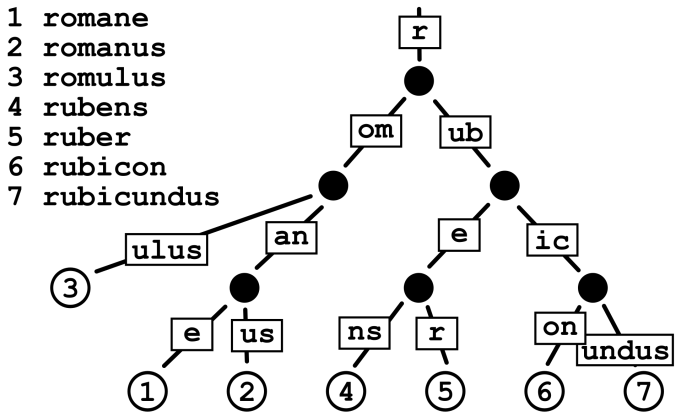
3.4.1 Recursive Length Prefix (RLP)

Το RLP είναι μια κωδικοποίηση που επιτρέπει τη σειριοποίηση μιας αυθαίρετης αναδρομικής δομής δεδομένων. Η ανάγκη της σειριοποίησης προκείπτει από την απαίτηση οι κόμβοι να επικοινωνούν αποδοτικά έτσι ώστε να ελαχιστοποιείται το latency στην ανταλλαγή μηνυμάτων καθώς επίσης και από την ανάγκη ο κάθε κόμβος να μπορεί να συμπυκνώσει όσο περισσότερο γίνεται την πληροφορία και να μπορέσει να συγκρατήσει όσο περισσότερα δεδομένα γίνεται. Ενδεικτικά, δείχνουμε μια σειροποίηση μιας απλής δενδρικής δομής:

$$[1, \text{"}dog\text{"}, [\text{"}cat\text{"}, [\text{"}rabbit\text{"}]]] \rightarrow 0xd283646f67cd83636174c88672616262697480$$

3.4.2 Radix Trees

Τα radix trees ή αλλιώς patricia trees είναι μια δενδρική δομή δεδομένων η οποία αποσκοπεί στην βελτιστοποίηση του χώρου αποθήκευσης. Στη δομή αυτή, η κοινή πληροφορία μεταξύ στοιχείων συμπυκνώνεται σε μια μοναδική αναπαράσταση, ενώ η υπόλοιπη πληροφορία αποτελεί ένα "παιδί" της κοινής. Έτσι, δημιουργείται ένα δέντρο στο οποίο κάθε μονοπάτι κατασκευάζει ένα στοιχείο. Η χωρική τους αποδοτικότητα είναι μεγαλύτερη όσο τα δεδομένα που περιέχουν έχουν πολλά κοινά τμήματα. Τα radix trees έχουν ευρεία εφαρμογή στα λεξικά καθώς επιτυγχάνουν γρήγορους χρόπους εισαγωγής και αναζήτησης.



Σχήμα 3.1: *Radix tree*

3.4.3 Merkle Trees

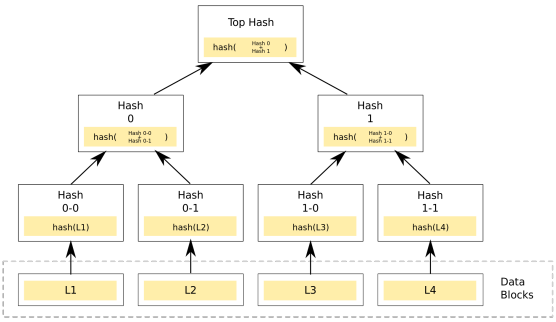
Τα Merkle trees είναι μια δενδρική δομή δεδομένων η οποία αποσκοπεί στην επικύρωση πληροφορίας. Κάθε φύλλο του δέντρου αντιστοιχεί σε ένα block της πληροφορίας που μας ενδιαφέρει και το περιεχόμενο αυτού είναι το hash του block αυτού. Οι υπόλοιποι ενδιάμεσοι κόμβοι περιέχουν το hash που προκύπτει αν συνδυαστεί το περιεχόμενο των κόμβων παιδιών τους και έτσι αναδρομικά το δέντρο κατασκευάζεται μέχρι τη ρίζα του. Η δομή αυτή μπορεί να χρησιμοποιηθεί για να βεβαιωθεί κανείς ότι η πληροφορία που κατέχει είναι έγκυρη, εφόσον γνωρίζει την τιμή της ρίζας του δέντρου. Έτσι μπορεί διαδοχικά να κατακερματίσει την πληροφορία που κατέχει σύμφωνα με τη δομή του δέντρου και αν καταλήξει στο αναμενόμενο hash, τότε μπορεί να είναι σίγουρος ότι η πληροφορία είναι η σωστή.

3.4.4 Merkle-Patricia Trees

Θα συμπληρωθεί σοον...

3.4.5 Block

Στο Ethereum, η ολική κατάσταση του συστήματος αποθηκεύεται σε blocks. Ένα block αποτελεί μια συλλογή δεδομένων, οπού εκεί περιέχονται το ιστορικό των συναλλαγών και διάφορα μεταδεδομένα για την κατάσταση του συστήματος. Τα μεταδεδομένα βρίσκονται στην κεφαλή του block με τις συναλλαγές να ακολουθούν σαν μια λίστα από hashes.



Σχήμα 3.2: *Merkle tree*

Το περιεχόμενο του header ενός block παρατίθεται στη συνέχεια με μια συνοπτική περιγραφή, ενώ πιο λεπτομερής ανάλυση των επιμέρους τμημάτων θα γίνει στα κατάλληλα σημεία στη συνέχεια της εργασίας.

Parent Hash Η Keccak-256 ηαση τιμή του block header του προηγούμενου block.

Ommers Hash Η Keccak-256 hash τιμή των ommer blocks για το συγκεκριμένο block.

Beneficiary Η διεύθυνση η οποία θα πάρει την ανταμοιβή για το επιτυχημένο mining του συγκεκριμένου block. Συνήθως, η διαύθυνση αυτή αντιστοιχεί στην διεύθυνση του miner του block.

State Root Η Keccak-256 hash τιμή της ρίζας του state δέντρου που αντιστοιχεί στο συγκεκριμένο block.

Transaction Root Η Keccak-256 hash τιμή της ρίζας του transactions δέντρου που αντιστοιχεί στο συγκεκριμένο block.

Receipts Root Η Keccak-256 hash τιμή της ρίζας του receipts δέντρου που αντιστοιχεί στο συγκεκριμένο block.

Logs Bloom Τα logs που προέκυψαν από την εκτέλεση των συναλλαγών για το συγκεκριμένο block.

Difficulty Η δυσκολία του συγκεκριμένου block.

Number Ο αύξων αριθμός του συγκεκριμένου block.

Gas Limit Ο μέγιστος αριθμός gas που μπορούσε να χρησιμοποιηθεί για το συγκεκριμένο block.

Gas Used Το gas που χρησιμοποιήθηκε για το συγκεκριμένο block.

Timestamp Ένα timestamp που αφορά τη στιγμή που αυτό το block ξεκίνησε.

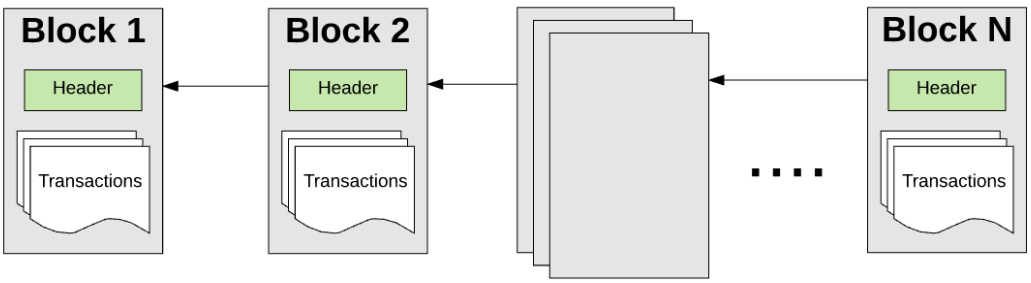
Extra Data Επιπλέον πληροφορία για το block. Πρόκειται για εναν 32-byte πίνακα.

Mix Hash Ένα hash που επιβεβαιώνει ότι το κατάλληλο ποσό υπολογισμών έχει γίνει για το συγκεκριμένο block.

Nonce Η τιμή που χρησιμοποιήθηκε για να γίνει το mining του συγκεκριμένου block.

Ommer Block Headers Οι κεφαλίδες των ommer blocks που χρησιμοποιήθηκαν σε αυτό το block.

Πολλά blocks που συνδέονται αλυσιδωτά κατασκευάζουν το blockchain. Όπως φαίνεται από τα παραπάνω, ένα block αναφέρεται απευθείας στο προηγούμενό του μέσω της Parent Hash τιμής που έχει στην κεφαλίδα του. Πέρα από την κεφαλίδα, το block περιέχει και μια λίστα από τα hashes των συναλλαγών όπου έλαβαν μέρος στο block αυτό. Όπως επίσης φάινεται, το ίδιο το block περιέχει όλη την πληροφορία για τις συναλλαγές που έγιναν, το νέο state που έφτιαξαν όσον αφορά τους λογαριασμούς που χρησιμοποιήθηκαν, καθώς και το σύνολο των μεταδεδομένων που μπορούν να χρησιμοποιηθούν από κάποιον χρήστη και αφορούν μια οπτική απεικόνιση των αποτελεσμάτων των συναλλαγών του block.

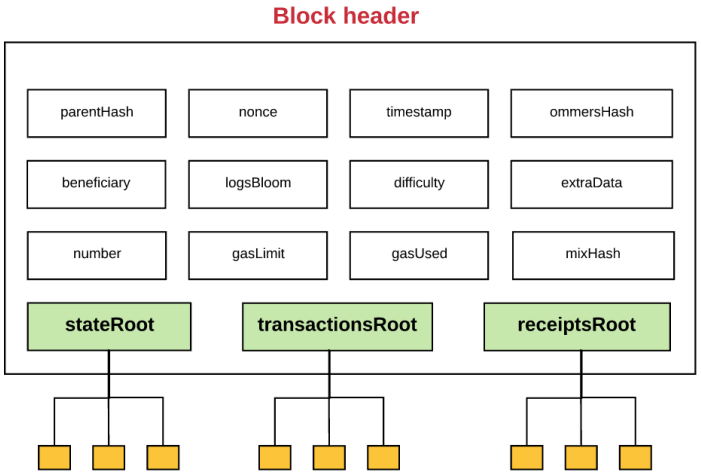


Εικόνα 3.1: *Ethereum blockchain*

Εδώ αξίζει να σημειωθεί ότι σε κάθε block αντιστοιχούν 3 Merkle-Patricia δέντρα για το state, τα transactions και τα receipts. Από τα δέντρα αυτά, το state δέντρο είναι κοινό για όλα τα blocks και το κάθε block ουσιαστικά μέσω του hash της ρίζας του αναπαριστά την αλλαγή του global state. Μόνο τα hashes των ριζών των δέντρων αυτών περιέχονται στα headers ενώ ένας κόμβος μπορεί να εκτελέσει querries για οποιαδήποτε συναλλαγή και state αφού χρησιμοποιεί μια βάση δεδομένων η οποία αποθηκεύει όλα τα ενδιάμεσα hashes και τις τιμες των φύλλων. Το Go-Ethereum χρησιμοποιεί leveldb για την αποθήκευση των δέντρων ώστε να πετυχαίνει γρήγορες αναζητήσεις.

Genesis block

Εδώ αξίζει να σημειώσουμε ότι κάθε blockchain στο Ethereum έχει ένα block αφετηρίας το οποίο ονομάζεται genesis block. Αυτό πέρα από το ότι εξυπηρετεί σαν block 0, ορίζει



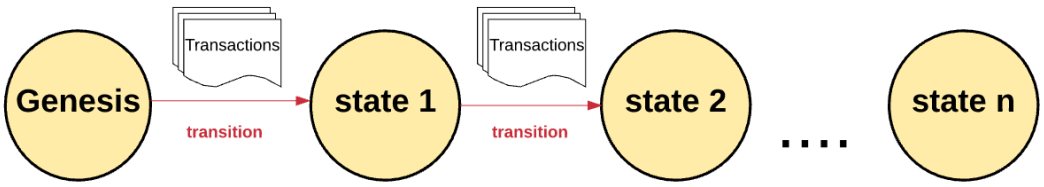
Εικόνα 3.2: *Ethereum block*

επίσης τις παραμέτρους με τις οποίες ξεκινάει να συμπεριφέρεται η αλυσίδα, όπως το id της, το difficulty του mining και τα block gas limits.

3.4.6 Συναλλαγές

Όπως έχουμε αναφερθεί, το Ethereum αναπαριστά έναν εικονικό υπολογιστή με ένα global state. Αυτό το state μπορεί να μεταβληθεί είτε από τις μεταφορές αξίας μεταξύ των λογαριασμών του δικτύου είτε από την εκτέλεση smart contracts. Ο μοναδικός τρόπος που ένας χρήστης μπορεί να αλλάξει το state του δικτύου είναι μέσω της εκτέλεσης συναλλαγών. Αυτές μπορούν να αφορούν είτε τη μεταφορά Ether από κάποιον λογαριασμό σε κάποιον άλλο λογαριασμό είτε την εκτέλεση ενός smart contract.

Στο Ethereum οι συναλλαγές δεν σταματάνε ποτέ. Όταν ένας λογαριασμός εκτελεί μια συναλλαγή τότε αυτή γνωστοποιείται αμέσως στο δίκτυο, ελέγχεται για την εγκυρότητά της και με βάση κάποια κριτήρια δρομολογείται για να αποθηκευτεί στο blockchain. Το σύνολο των συναλλαγών που εκτελούνται μέσα σε ένα block θα αλλάξει το state της αλυσίδας και θα δημιουργήσει το επομένο.



Σχήμα 3.3: *Οι συναλλαγές αλλάζουν το state*

Μια συναλλαγή αποτελείται από τα εξής δεδομένα:

Nonce Ένας αύξων ακέραιος που εκφράζει το πλήθος των συναλλαγών που έχουν εκτελεστεί επιτυχώς από κάποιον λογαριασμό.

Gas Price Η τιμή σε Wei που ο λογαριασμός είναι διατεθιμένος να πληρώσει στο δίκτυο ανά μονάδα gas έτσι ώστε η συγκεκριμένη συναλλαγή να εκτελεστεί.

Gas Limit Το μέγιστο ποσό gas που ο λογαριασμός είναι διατεθιμένος να χρησιμοποιήσει ώστε η συναλλαγή του να εκτελεστεί.

To Η διεύθυνση του παραλήπτη της συναλλαγής.

Value Το ποσό σε Wei που ο χρήστης επιθυμεί να μεταφέρει στον παραλήπτη της συναλλαγής.

v, r, s Τα κρυπτογραφικά κλειδιά που εξασφαλίζουν την εγκυρότητα της συναλλαγής.

Ένα block δεν μπορεί να περιέχει ένα αυθαίρετο αριθμό συναλλαγών καθώς σε μια τέτοια περίπτωση ο ανταγωνισμός μεταξυ των miners θα τους οδηγούσε να συλλέγουν όσο το δυνατόν περισσότερες συναλλαγές για να μεγιστοποιήσουν το κέρδος τους και έτσι το δίκτυο θα καθυστερούσε. Επίσης, μια συναλλαγή δεν μπορεί να εκτελείται για πάντα, αφού έτσι το δίκτυο θα ήταν εκτεθειμένο σε κακόβουλες επιθέσεις. Κάποιος θα μπορούσε να δημιουργήσει ένα smart contract με έναν ατέρμων βρόχο και να μπλοκάρει όλους τους κόμβους όταν αυτοί το εκτελέσουν.

Έτσι, το gas limit της συναλλαγής θέτει ένα άνω φράγμα στους πόρους που μπορεί να αυτή καταλάβει. Συνοπτικά, το gas αποτελεί μια μονάδα μέτρησης πόρων και είναι απαραίτητο για να διασφαλιστεί η συνεχής λειτουργία του συστήματος. Μια απλή μεταφορά Ether μεταξύ λογαριασμών καταναλώνει ένα σταθερό ποσό gas. Ανάλογα με το gas price που έχει θέσει ο χρήστης αυτή η συναλλαγή αποκτά μια προτεραιότητα και δίνει κίνητρο στους miners να τη συμπεριλάβουν στο block που κάνουν mine. Μπορεί κανείς να το φανταστεί σαν μια δωροδοκία προς τον miner ώστε να εξασφαλιστεί η ταχύτητα της συναλλαγής.

Αντίστοιχα, και ένα contract καταναλώνει πόρους, άρα gas, το οποίο δεν είναι σταθερό αλλά εξαρτάται άμεσα από το είδος του κώδικα που βρίσκεται στο smart contract. Ειδικότερα, θα εξετάσουμε τον τρόπο που το gas καταναλώνεται στο αντίστοιχο κεφάλαιο των smart contracts, ωστόσο αξίζει να αναφερθεί ότι αν το gas που καταναλώνει ο miner για την εκτέλεση του contract υπερβεί το gas limit που ο χρήστης έχει ορίσει, τότε η συναλλαγή θεωρείται αυτομάτως αποτυχημένη και δεν προκαλεί καμιά μεταβολή στο global state του δικτύου.

3.5 Smart Contracts

Ο ορισμός διατυπώνει ότι τα smart contracts δεν είναι τίποτα άλλο πέρα από προγράμματα γραμμένα σε κάποια high-level γλώσσα προγραμματισμού, τα οποία εκτελούνται πάνω στο blockchain που υπάγονται, μεταβάλλοντας το state του. Το κίνητρο για την χρησιμοποιήσή τους είναι η επέκταση των δυνατοτήτων του blockchain είτε σε επίπεδο αυτοματοποιήσης

συναλλαγών, είτε σε υλοποίηση πολύπλοκων εφαρμογών που συνδυάζουν δεδομένα εισόδου από τον χρήστη μαζί με το state του blockchain τη στιγμή της εκτέλεσής τους.

Όσον αφορά τη χρήση τους στο Ethereum, ιδιαίτερο ενδιαφέρον παρουσίαζει ο τρόπος με τον οποίο το σύστημα επιτυγχάνει την εκτέλεση τους εξασφαλίζοντας την ατομικότητα, καθώς επίσης και ο τρόπος με τον οποίο το σύστημα δεσμεύει πόρους και κεφάλαια ώστε κακόβουλα smart contracts να μην επηρεάζουν τη λειτουργία του από άποψη διαθεσιμότητας(availability).

3.5.1 Συγγραφή των Smart Contracts

Υπάρχουν δύο κύριες γλώσσες προγραμματισμού στις οποίες μπορεί κανείς να γράψει smart contracts προς εκτέλεση στο Ethereum και αυτές είναι οι Solidity και η Vyper, ωστόσο η πιό συχνά εμφανιζόμενη είναι η Solidity. Πρόκειται για μια αντικειμενοστρεφή γλώσσα προγραμματισμού η οποία παρουσιάζει πολλές ομοιότητες με τις πιο γνωστές αντικειμενοστρεφείς γλώσσες και έχει μεγάλη ομοιότητα με την Javascript. Παρακάτω παρουσιάζεται ένα απλό δείγμα κώδικα, το οποίο χρησιμοποιούμε και αργότερα στην εργασία, το οποίο εκτελεί εισαγωγή και εύρεση στοιχείων με βάση κάποιο κλειδί, το οποίο ο χρήστης στέλνει μαζί με την εντολή εκτέλεσης:

3.5.2 Δημιουργία ενός smart contract

Αφού γίνει η συγγραφή ενός smart contract, ο χρήστης πρέπει να κάνει την κατάλληλη μεταγλώτιση και να το μετατρέψει σε bytecode μορφή την οποία μπορεί να αντιληφθεί το EVM που θα αναλάβει την εκτέλεσή του. Το αποτέλεσμα θα είναι μια σειριοποιημένη μορφή του κώδικα και ο χρήστης θα πρέπει να εκτελέσει μια συναλλαγή για να δημιουργήσει το contract στο δίκτυο.

Η συναλλαγή δημιουργίας του contract δεν διαφέρει από μια απλή συναλλαγή με τη διαφορά ότι έχει ένα επιπλέον πεδίο, το οποίο περιέχει τον κώδικα σε μορφή bytecode. Αυτός ο κώδικας θα πρέπει να αποθηκευτεί στο δίκτυο, επομένως προκύπτει η ανάγκη ο χρήστης να διαθέσει κάποια Wei ώστε να το αποζημιώσει για τους πόρους που θα διαθέσει. Για την εναπόθεση του contract στο δίκτυο, ο χρήστης πρέπει να στείλει μια συναλλαγή χωρίς παραλήπτη, με όλα τα υπόλοιπα πεδία συμπεριλαμβανομένου και του πεδίου data με το bytecode, και το δίκτυ θα το εκλάβει σαν εντολή για τη δημιοργία του contract.

Στη συνέχεια, το δίκτυο θα κατασκευάσει ένα λογαριασμό για το contract μέσω του οποίου ο χρήστης θα αλληλεπιδρά με αυτό. Όντας λογαριασμός, ένα contract μπορεί να κατέχει δικά του Ether και μπορεί να τα διαχειριστεί όπως ο κώδικάς του το επιτρέπει. Τα Ether αυτά μεταφέρονται στο contract όπως ο χρήστης θα εκτελούσε οποιαδήποτε άλλη απλή συναλλαγή.

3.5.3 Μνήμη

Τα smart contracts στην Solidity υποστηρίζουν δομές δεδομένων όπως απλές μεταβλητές, λίστες, maps. Επομένως, συμπεραίνουμε ότι εκτός από στατικές, οι δομές μπορεί να είναι και δυναμικές άρα οι απαιτήσεις σε μνήμη από ένα contract δεν μπορεί να είναι συ-

γκεκριμένες ή προκαθορισμένες. Το Ethereum κοστολογεί τον χώρο αυτό και χρεώνει τον εκτελεστή της κάθε συναλλαγής με το contract ανάλογα με το χώρο που θα χρησιμοποιήσει.

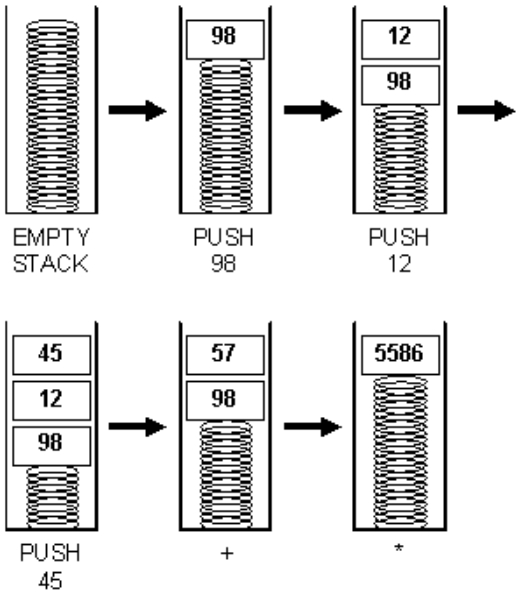
Ο τρόπος κοστολόγησης είναι τέτοιος έτσι ώστε να γίνεται ανάλογα με το μικρότερο πολλαπλάσιο των 32 bytes που απαιτούνται για να μπορέσει να διευθυνσιοδοτηθεί κάθε κομμάτι μνήμης που ανήκει στο πρόγραμμα. Το κίνητρο πίσω από αυτή την επιλογή είναι οι προγραμματιστές να βελτιστοποιούν τα προγράμματα έτσι ώστε να μην κάνουν αλόγιστη χρήση των πόρων του συστήματος.

3.5.4 EVM

Τα smart contracts όπως είδαμε, είναι μεταγλωτισμένος κώδικας ο οποίος αποθηκεύεται στο δίκτυο και μπορεί να εκτελεστεί κατά απαίτηση μιας συναλλαγής. Ωστόσο, το γεγονός ότι αποθηκεύονται σε μια συγκεκριμένη μορφή κοινή για όλους τους κόμβους σημαίνει ότι θα πρέπει να μπορούν να εκτελεστούν από κάθε κόμβο, ανεξάρτητα από το λειτουργικό του σύστημα και την αρχιτεκτονική που ακολουθεί το hardware του. Για το σκοπό αυτό, το Ethereum ανέπτυξε το EVM.

Το EVM αποτελεί μια εικονική μηχανή που μπορεί και εκτελεί bytecode κώδικα που έχει προκύψει από μεταγλώτιση των υποστηριζόμενων γλωσσών για τα smart contracts. Πρόκειται για ένα stack-based μηχάνημα που δεν ακολουθεί την κλασσικη von Neumann αρχιτεκτονική αλλά αποθηκεύει τον κώδικα σε μια εικονική μνήμη ROM με την οποία μπορεί να αλληλεπιδράσει μέσω ειδικών εντολών.

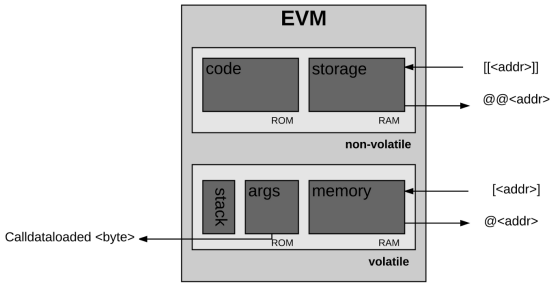
Το runtime έχει έχει μια στοίβα μεγέθους 1024 στοιχείων με το κάθε στοιχείο να έχει μέγεθος 256 bit. Κατά την εκτέλεση του προγράμματος η στοίβα επηρεάζεται όπως στο runtime άλλων γλωσσών προγραμματισμού, ενώ αν γεμίσει τότε το EVM αδυνατεί να τερματίσει την εκτέλεση και η συναλλαγή αποτυγχάνει.



Εικόνα 3.3: *EVM stack*

EVM Runtime

Όταν πρόκειται να εκτελέσει ένα contract, το EVM δημιουργεί ένα περιβάλλον στο οποίο το οποίο έχει σαν στατε την πιο προσφατη κατάσταση του συστήματος, δηλαδή το state του τελευταίου block. Επίσης, το EVM κρατάει σαν άλλα δεδομένα το gas που έχει μείνει και μπορεί να χρησιμοποιηθεί για την δεδομένη συναλλαγή, τον κώδικα που θα εκτελέσει στην περίπτωση που πρόκειται για smart contract, διάφορες σημαντικές διευθύνσεις κλπ.



Εικόνα 3.4: *EVM Runtime*

Όταν ξεκινάει να εκτελείται μια συνάρτηση ενός smart contract, το EVM εκτελεί διαδοχικά κάθε OPCODE από το contract ακολουθώντας την λογική του. Ανάλογα με την εντολή που εκτελεί αφαιρείται το αντίστοιχο gas από τη συναλλαγή που ξεκίνησε αυτή την εκτέλεση εώς ότου το πρόγραμμα τερματίσει ή τελειώσει το gas. Στην περίπτωση που τελειώσει το gas ή το πρόγραμμα οδηγηθεί σε κάποια εξαίρεση, τότε η εκτέλεση αυτομάτως αποτυγχάνει και επομένως και η ίδια η συναλλαγή. Σε διαφορετική περίπτωση, η συναλλαγή πετυχαίνει και το gas που χρησιμοποιήθηκε αφαιρείται από το gas limit της συναλλαγής και η αξία του επιστρέφεται σαν αποζημίωση στον λογαριασμό που έστειλε την συναλλαγή. Επομένως, είναι εύλογο το γεγονός ότι το EVM απαιτεί ο λογαριασμός να διαθέτει εξαρχής τα Ether που αντιστοιχούν στο gas limit που αυτός έχει ορίσει.

Επικοινωνία μεταξύ Smart Contracts

Η εκτέλεση των smart contracts δεν περιορίζεται μόνο στην απλή λήψη συναλλαγών και εκτέλεσης κώδικα. Ο κώδικας ενός smart contract μπορεί να εμπεριέχει και την κλήση κάποιου άλλου smart contract που είναι επίσης στο δίκτυο. Έτσι, η λειτουργικότητα των συμβολαίων μπορεί να επεκταθεί συνολικά και οι χρήστες να μπορούν να φτιάξουν εφαρμογές που εκτελούν πολύπλοκες διαδικασίες που συνδυάζουν πολλές διαφορετικές λογικές. Χαρακτηριστικό παράδειγμα είναι τα αποκετρωμένα ανταλλακτήρια ή DeX, τα οποία είναι επί της ουσίας κώδικας που συνδυαζόμενος με τον κώδικα των Token Contracts επιτρέπει στους χρήστες να συναλλάσσονται tokens.

Σε επίπεδο EVM, όταν ένα contract καλεί κάποιο άλλο, τότε ο Ε΄Μ εκτελεί κανονικά τη συνάρτηση που κλήθηκε. Για την εκτέλεση κάθε εντολής του νέου contract χρησιμοποιείται το gas που δόθηκε στην αρχική συναλλαγή, επομένως είναι δυνατό αυτό να τελειώσει και να αποτύχει η κλήση του contract. Στην περίπτωση αυτή ολόκληρη η συναλλαγή θεωρείται αποτυχημένη. Σε αντίθετη περίπτωση, η κλήση πετυχαίνει και ο EVM επιστρέφει στην αρχική

εκτέλεση του πρώτου contract. Αν υπάρχει περισσευούμενο gas από μια τέτοια εκτέλεση, τότε αυτό επιστρέφεται στον λογαριασμό που εκτέλεσε τη συναλλαγή.

3.5.5 Mining

Το mining αποτελεί την καρδία ενός blockchain συστήματος και είναι ο κύριος τρόπος για να διασφαλιστεί η ακαιρεότητά του. Μέσω του mining, οι ειδικοί κόμβοι που ονομάζονται miners, δημιουργούν νέα blocks συναλλαγών με σκοπό αυτά να συμπληρώσουν την αλυσίδα και να συνεχιστεί η λειτουργία του δικτύου. Στη βιβλιογραφία, βλέπουμε ότι το mining αποτελεί έναν consensus μηχανισμό έτσι ωστε όλοι οι κόμβοι να κατέχουν μια σωστή και κοινή έκδοση του blockchain. Υπάρχουν πολλοί consensus μηχανισμοί, όμως το Ethereum στην προκειμένη περίπτωση χρησιμοποιεί το Proof of Work, το οποίο θα αναλυθεί στη συνέχεια.

Proof of Work

Το Proof of Work αποτελεί μια μορφή για κρυπτογραφική απόδειξη ότι έχει γίνει κάποια προσπάθεια προκειμένου να εκτελεστεί μια εργασία. Ο σκοπός του είναι να αποφευχθεί η αλόγιστη εκμετάλλευση μια υπηρεσίας καθώς αν χρησιμοποιηθεί, τότε ο κάθε χρήστης θα πρέπει να καταναλώσει και δικούς του πόρους προκειμένου να επιφέρει κάποια κακόβουλη επίπτωση στο σύστημα.

Στο Ethereum, το Proof of Work δεν είναι τίποτα άλλο πέρα από μια απλή υπόθεση μιας τιμής προκειμένου να εξασφαλιστεί ένα συγκεκριμένο αποτέλεσμα. Πιο συγκεκριμένα, οι miners, αφού έχουν κατασκευάσει το block θα πρέπει να υπολογίσουν μια τιμή, που ονομάζεται nonce, προκειμένου να τη συνδυάσουν με το block και αφού υπολογίσουν το hash που προκύπτει να βεβαιωθούν ότι η τιμή αυτή είναι μικρότερη από κάποια τιμή που έχει ορίσει το σύστημα.

Η τιμή δυσκολίας που ορίζει το σύστημα έχει άμεση συσχέτιση με τον επιθυμητό χρόνο για την ένταξη ενός block στην αλυσίδα. Όταν κάποιος miner καταφέρνει να υπολογίσει το κατάλληλο nonce πιο γρήγορα από τον αναμενόμενο χρόνο, τότε η δυσκολία του συστήματος θα αυξηθεί προκειμένου να επέλθει ισορροπία, ενώ αντίστοιχα αν κάποιο block αργήσει να γίνει mined, τότε η δυσκολία θα μειωθεί. Στο Ethereum ο στόχος είναι το κάθε block να γίνεται mine κάθε 15 δευτερόλεπτα.

$$n \leq \frac{2^{256}}{H_d}$$

Όταν κάποιος miner βρει το κατάλληλο nonce, τότε γνωστοποιεί στο δίκτυο το νέο βλοσκ που κατασκεύασε και μετά το δίκτυο πρέπει να αποφασίσει αν θα το εντάξει στο blockchain. Έτσι, ο κάθε κόμβος επικυρώνει τις συναλλαγές που περιέχονται στο block και στη συνέχεια ελέγχει ότι το hash που προκύπτει αν συνδυαστεί με το nonce που συμπεριλήφθηκε οδηγεί στο αναμενόμενο hash στόχο. Έτσι, όλοι οι κόβοι ψηφίζουν και αν επιτευχθεί συμφωνία το νέα block ενσωματώνεται στην αλυσίδα.

Επεκτασιμότητα

Η επίτευξη του consensus μέσω του δικτύου του Ethereum απαιτεί οι κόμβοι να ανταλλάσσουν συνεχώς μεταξύ τους όλες τις συναλλαγές που λαμβάνουν. Επιπλέον, όταν καλείται να επικυρωθεί ένα νέο block, οι συναλλαγές θα πρέπει να μεταδοθούν και πάλι μέσα στο block. Αυτό έχει σαν αποτέλεσμα ότι όσο περισσότεροι είναι κόμβοι που συμμετέχουν στο δίκτυο, τόσο αυτό να πάσχει από πρόβλημα επικοινωνίας. Έτσι, το δίκτυο δεν είναι εύκολα επεκτάσιμο και η συμμετοχή νέων κόμβων πάνω από κάποιο όριο δημιουργεί bottleneck επικοινωνίας στο δίκτυο.

Forking

Ο κύριος λόγος για τον οποίο χρειάζεται το Proof Of Work είναι το πρόβλημα που μπορεί να προκύψει αν παράγονται συνεχώς νέα blocks. Σε μια τέτοια περίπτωση, δύο κόμβοι μπορεί να λάβουν ταυτόχρονα από ένα block που αντιστοιχεί στην επόμενη θέση της αλυσίδας αλλά τα δεδομένα του καθενός να είναι διαφορετικά μεταξύ τους. Αν υποθέσουμε ότι και τα δύο blocks έχουν έγκυρες συναλλαγές, τότε ο κάθε κόμβος θα τα εντάξει στο τοπικό του αρχείο αλλά στην πραγματικότητα οι δύο κόμβοι θα έχουν διαφορετική αλυσίδα. Έτσι προκύπτει το λεγόμενο forking ή κατακερματισμός του δικτύου.

Οι χρόνοι για το mining στο Ethereum είναι όπως αναφέραμε προηγουμένως 15 δευτερόλεπτα και αυτός είναι ένας οριακός χρόνος για να αποφευχθεί το forking. Όταν αυτό συμβαίνει, προκύπτουν 2 δίκτυα τα οποία προχωρούν παράλληλα και είναι πιθανό σε καθένα από αυτά να ισχύουν διαφορετικές καταστάσεις. Έτσι, το δίκτυο ακολουθεί το πρωτόκολλο GHOST(Greedy Heaviest Observed Subtree), το οποίο πολύ απλά υπαγορεύει ότι σε περίπτωση forking, τότε το δίκτυο θα επιλέξει σαν έγκυρη την αλυσίδα που έχει το περισσότερο Proof of Work, δηλαδή με άλλα λόγια την μεγαλύτερη αλυσίδα. Οι συναλλαγές που έχουν επιβεβαιωθεί για τα άλλα κλαδιά αυτομάτως απορρίπτονται.

Ανταμοιβές

Ένας miner ανταμοίβεται όταν το block του έχει συμπεριληφθεί στο blockchain, ωστόσο υπάρχει η πιθανότητα να έχει πράγματι καταβάλει το κατάλληλο ποσό δουλειάς και να έχει έντιμα τοποθετήσει τις συναλλαγές μέσα στο block αλλά το mined block εν τέλει να απορριφθεί από το δίκτυο. Για το λόγο αυτό, ένα τέτοιο block ονομάζεται ommer block και επρόκειτο να ανταμοιφθεί όταν γίνει η διανομή των κερδών στους miners, ωστόσο θα λάβει ένα αρκετά μικρότερο ποσό σε σχέση με τον miner που επιλέθηκε. Αυτό γίνεται έτσι ώστε οι miners να έχουν το κίνητρο να συμμετέχουν στη διαδικασία ακόμα και αν δεν επιλεγούν από το GHOST.

Αυτό γίνεται από τους μελλοντικούς miners οι οποίοι εντοπίζουν τέτοια blocks και τα συμπεριλαμβάνουν στην κεφαλίδα του νέου block που κάνουν mine. Προκειμένου ένα miner που έφτιαξε κάποιο ommer block να συμπεριληφθεί στο νέο block από κάποιον miner, θα πρέπει το ommer block του να μην είναι παλαιότερο από 6 blocks.

Κεφάλαιο 4

Solana

Το Solana είναι ένα νέου είδους blockchain που ξεκίνησε σαν ιδέα το 2017 από τον Anatoly Yakovenko, ο οποίος παρουσίασε το whitepaper [5] στο οποίο κάνει μια περιγραφή της δομής του. Σκοπός του Solana είναι να αποτελέσει ένα δίκτυο που επιτρέπει πολύ γρήγορες συναλλαγές παρέχοντας ταυτόχρονα και την ασφάλεια που απαιτείται από ένα blockchain.

Για το σκοπό αυτό, έχουν εισαχθεί πολλές καινοτομίες στο σύστημα όπως τα Proofs of History, Stake και Replication και ακολουθούνται κανόνες που δίνουν οικονομικά κίνητρα και αντικίνητρα στους χρήστες έτσι ώστε να εξασφαλίζεται η ακεραιότητα. Ασφαλώς, το δίκτυο επιτρέπει τη χρήση smart contracts τα οποία ονομάζονται πλέον programs. Τέλος, ο τρόπος με τον οποίο μπορεί να αλληλεπιδράσει κάποιος με το σύστημα διαφέρει πάρα πολύ από τον τρόπο που μπορεί να το κάνει μέσω των άλλων blockchains.

Η γλώσσα που χρησιμοποιήθηκε για την ανάπτυξη του συστήματος είναι η Rust, μια προστακτική γλώσσα προγραμματισμού η οποία δίνει το πλεονέκτημα ότι έχει πολύ γρήγορο runtime σύστημα. Αυτό συμβαίνει καθώς χρησιμοποιεί ένα πρωτότυπο borrow-checking μηχανισμό, ο οποίος δεν αφήνει τον προγραμματιστή να αφήσει δεσμευμένους χώρους μνήμης στο πρόγραμμα χωρίς να αναφέρεται σε αυτούς (dangling references) με αποτέλεσμα να μην είναι απαραίτητη η χρήση garbage collector. Επίσης, η Ρυστ επιτρέπει πάρα πολλές low-level λειτουργικότητες παρόμοιες με τη C και C++ με αποτέλεσμα τα προγράμματα να μπορούν να είναι βελτιστοποιημένα.

Στο κεφάλαιο αυτό, θα κάνουμε μια σύντομη αναφορά στους λογαριασμούς και τις συναλλαγές στο σύστημα, ενώ θα αναλύσουμε σε βάθος άλλες πλευρές του όπως τη δομή και οι ιδιότητες των κόμβων, τα Proof of History, Stake και Replication καθώς και το runtime του συστήματος.

4.1 Βασικές οντότητες

Σε αυτό το μέρος θα εξετάσουμε πώς διαρθρώνονται οι λογαριασμοί στο Solana και πώς ο χρήστης μπορεί να εκτελέσει συναλλαγές είτε αυτές είναι απλές μεταφορές αξίας, είτε είναι κλήσεις προγραμμάτων.

Στο Solana τα πάντα είναι προγράμματα και οι λογαριασμοί επί της ουσίας υπόκεινται στην ιδιοκτησία από κάποια προγράμματα. Θα αναλύσουμε αυτό το μοντέλο σε αυτό το κεφάλαιο.

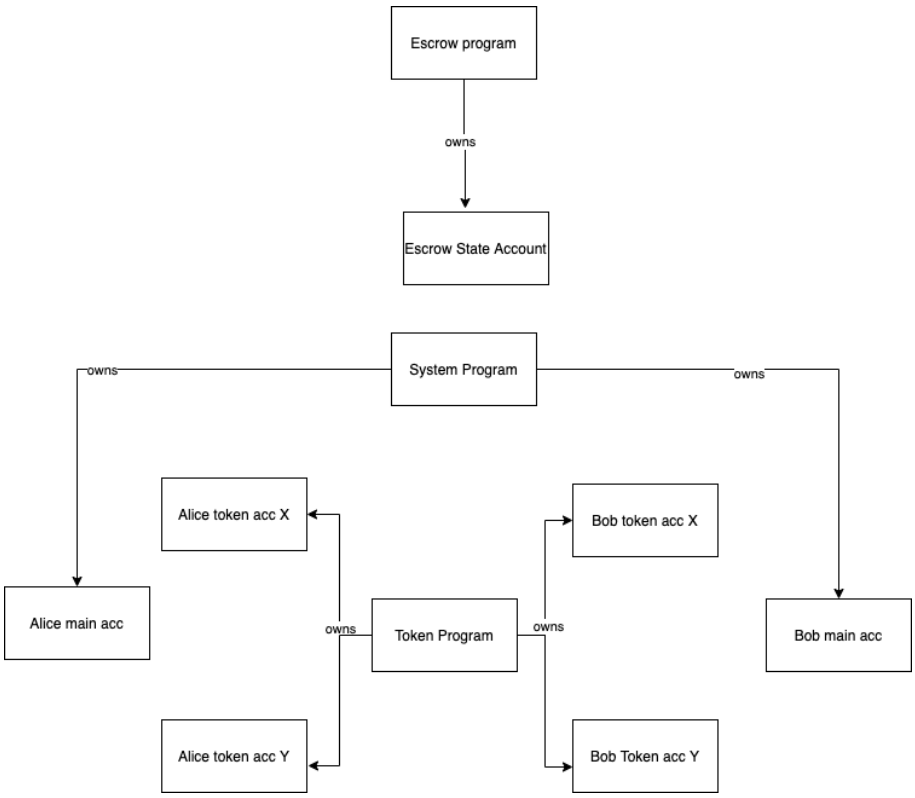
4.1.1 Λογαριασμοί

Ένας λογαριασμός αποτελεί μια οντότητα η οποία μπορεί να έχει στην κατοχή της δεδομένα και SOL, το επίσημο νόμισμα του Solana. Επίσης, ένας λογαριασμός ανήκει στο System Program, το οποίο θα αναλυθεί στη συνέχεια, και μπορεί να χρησιμοποιηθεί από όποιον έχει την κυριότητά του.

Υπάρχουν πολλά είδη λογαριασμών, όπως ο Data Account όπου είναι ένας λογαριασμός που σκοπός του είναι να αποθηκεύει δεδομένα. Τα περιεχόμενα του λογαριασμού μπορούν να διαβάζονται και να μεταβάλλονται από προγράμματα στο δίκτυο. Άλλα είδη λογαριασμών είναι το Program Account, Vote Account, Token Account.

Ένας λογαριασμός έχει κάποιον ιδιοκτήτη. Οι απλοί λογαριασμοί που χρησιμοποιούνται για τις συναλλαγές έχουν σαν ιδιοκτήτη το System Program. Η ροή των εντολών για μια απλή μεταφορά SOL μεταξύ δύο λογαριασμών εμπεριέχει την εντολή στο System Program να εκτελέσει τη μεταφορά μέσω του κλειδιού του αποστολέα προκειμένου να επιβεβαιωθεί ότι μόνο ο πραγματικός κάτοχος μπορεί να εκτελέσει τη μεταφορά.

Γενικά, οι λογαριασμοί αποτελούν ένα αρκετά δυσνόητο κεφάλαιο στο Solana. Ενδεικτικά παραθέτουμε ένα σχήμα που δείχνει τις σχέσεις μεταξύ των απαραίτητων λογαριασμών για τη δημιουργία ενός κλασσικού escrow προγράμματος με σκοπό οι χρήστες Bob και Alice να ανταλλάξουν μεταξύ τους τα tokens X και Y.



Σχήμα 4.1: *Λογαριασμοί για το escrow πρόγραμμα*

Εδώ έχουμε τους βασικούς λογαριασμούς του Bob και της Alice. Επιπλέον, έχουμε τους λογαριασμούς των tokens για τον καθέναν τους και άλλον έναν λογαριασμό που έχει τα δεδομένα του Escrow προγράμματος. Βλέπουμε ότι το System Program κατέχει τους

λογαριασμούς του Bob και της Alice, ενώ το Token Program κατέχει τους λογαριασμούς των tokens.

4.1.2 Προγράμματα

Σαν πρόγραμμα ονομάζουμε ένα smart contract που ζει στο Solana. Τα προγράμματα στο Solana έχουν άλλη δομή από αυτά στο Ethereum και είναι γραμμένα σε γλώσσες που υποστηρίζονται από το llvm. Συνήθως γράφορνται σε Rust, ωστόσο μπορεί να είναι υλοποιημένα σε C και C++.

Για να γίνει το deployment ενός προγράμματος, ο χρήστης πρέπει πρώτα να κάνει την απαραίτητη μεταγλώτιση τοπικά και να ανεβάσει το αποτέλεσμα της μεταγλώτισης στο δίκτυο δημιουργώντας έναν νέο λογαριασμό για το πρόγραμμα. Ο λογαριασμός αυτός έχει την επισήμανση του εκτελέσιμου.

Στη συνέχεια, ο χρήστης στέλνει την επιθυμητή εντολή στο πρόγραμμα και αυτή εκτελείται. Η κύρια διαφορά με τα smart contracts στο Ethereum είναι ότι στο Solana τα προγράμματα έχουν ένα σημείο εισόδου και είναι ευθύνη του προγράμματος να κατευθύνει τη ροή του βάσει των δεδομένων εισόδου από τον προγραμματιστή. Τα δεδομένα εισόδου λέγονται instruction data.

Εκτός από τα προγράμματα που ανεβάζουν οι προγραμματιστές, το σύστημα του Solana έχει μερικά native προγράμματα που εκτελούν βασικές λειτουργίες. Μερικά από αυτά είναι:

System Program Είναι αρμόδιο για τη δημιουργία λογαριασμών, την ανάθεση κυριότητας προγραμμάτων σε λογαριασμούς, τη μεταφορά SOL μεταξύ λογαριασμών και τη διαχείριση των fees για την εκτέλεση των συναλλαγών. Τα fees έχουν το ίδιο νόημα με τα fees στο Ethereum.

Config Program Θέτει διάφορες παραμέτρους του συστήματος.

Stake Program Είναι αρμόδιο για την διαδικασία του staking και τις ανταμοιβές στους ¨χορηγούς¨ των validators.

Vote Program Εκτελεί τη διαδικασία του voting για την επίτευξη του consensus.

BPF Loader Εναποθέτει, αναβαθμίζει και εκτελεί τα προγράμματα στην αλυσίδα.

Ed25519 Program Χρησιμοποιείται για την επεξεργασία των εντολών και την επικύρωση των υπογραφών σε αυτές.

Υπάρχουν και άλλα προγράμματα στο δίκτυο όπως το Token Program, Token Swap Program, Shared memory Program.

Program Derived Addresses (PDAs)

Τα PDAs αποτελούν διευθύνσεις στο σύστημα στις οποίες δεν αντιστοιχούν ιδιωτικά κλειδιά. Ωστόσο, αυτές οι διευθύνσεις μπορούν να χρησιμοποιηθούν από προγράμματα τα οποία με τη σειρά τους μπορούν να εκτελέσουν άλλες εντολές με τις διευθύνσεις αυτές. Ένα PDA δεν μπορεί να χρησιμοποιηθεί από έναν εξωτερικό παρατηρητή του συστήματος.

Η αναγκαιότητα ύπαρξης των PDAs προέρχεται από τις εξαρτήσεις που εισάγει το Solana μεταξύ των λογαριασμών του. Για παράδειγμα, ένα λογαριασμός που περιέχει ένα πρόγραμμα έχει αποθηκευμένη μόνο τη μεταγλωτισμένη εκδοχή του προγράμματος. Ο χώρος μνήμης που δεσμεύει δεν μπορεί να χρησιμοποιηθεί για κάποιο άλλο σκοπό. Κατά συνέπεια, αν το πρόγραμμα σκοπεύει να μεταβάλλει κάποια κατάσταση αυτή θα πρέπει να είναι αποθηκευμένη σε σειριοποιημένη μορφή σε κάποιον άλλο λογαριασμό που είναι αφιερωμένος στην αποθήκευση δεδομένων.

Ένα PDA μπορεί να δημιουργηθεί χρησιμοποιώντας τη διεύθυνσης του προγράμματος και έναν πίνακα από τιμές, ο οποίος μπορεί να είναι όποιος ο χρήστης επιθυμεί. Οι τιμές αυτές ονομάζονται seeds.

4.1.3 Συναλλαγές

Όπως και σε όλα τα blockchains, οι συναλλαγές είναι ο τρόπος με τον οποίο μπορεί να μεταβληθεί η κατάσταση του δικτύου. Μια συναλλαγή μπορεί να αφορά απλά τη μεταφορά ενός ποσού ΣΟΛ μεταξύ δύο λογαριασμών ή την εκτέλεση ενός προγράμματος.

Σε αντίθεση με το Ethereum, εδώ μια συναλλαγή αποτελείται από πολλές εντολές. Αυτές οι εντολές είναι υποσυναλλαγές ο οποίες θα μπορούσαν και να εκτελεστούν αυτόνομα. Γενικά, το runtime θα εκτελέσει μια συναλλαγή ατομικά, δηλαδή θα εκτελέσει όλες τις εντολές που περιλαμβάνονται σε αυτή σαν μία και η αποτυχία κάποιας από τις εντολές θα σημάνει και την αποτυχία της συναλλαγής.

Η δομή των εντολών στο Solana όταν αφορά την εκτέλεση κάποιου προγράμματος είναι ιδιαίτερη. Πέρα από τα δεδομένα εισόδου για την εκτέλεση του προγράμματος, ο χρήστης θα πρέπει να συμπεριλάβει στην εντολή και τους λογαριασμούς που θα χρησιμοποιηθούν και στην περίπτωση που κάποιος από αυτούς επρόκειτο να αλλάξει κατάσταση ή να εκτελέσει μια ενέργεια που απαιτεί την κυριότητά του, όπως η δημιουργία ενός άλλου λογαρισμού, τότε θα πρέπει να συμπεριληφθεί και η αντίστοιχη υπογραφή. Ενδεικτικά, παραθέτουμε μια εκτέλεση της εντολής get για το πρόγραμμα kvstore που θα χρησιμοποιηθεί στο πειραματικό κομμάτι της εργασίας. Έχουμε επιλέξει την γλώσσα Javascript και τη βιβλιοθήκη @solana/web3.js λόγω της εκφραστικότητάς της.

Στη συγκεκριμένη εντολή χρησιμοποιούμε 3 λογαριασμούς εκ των οποίων μόνο ο πρώτος υπογράφει τη συναλλαγή. Ο πρώτος λογαριασμός μιας εντολής είναι αυτός που πληρώνει για την εκτέλεσή της.

4.2 Proof Of History

Το Proof of History είναι μια κρυπτογραφική ακολουθία υπολογισμών που μπορεί να επιβεβαιώσει ότι έχει περάσει πραγματικός χρόνος μεταξύ των γεγονότων που περιλαμβάνει.

Συνήθως, τα άλλα blockchains δεν έχουν αυστηρές εκτιμήσεις του χρόνου και οι συναλλαγές σε ένα block που γίνεται ταυτόχρονα mine από 2 miners μπορεί να διαφέρουν σε σειρά ή και σε περιεχόμενο.

Υποθέτουμε ότι έχουμε ένα αρχικό γεγονός και επιλέγουμε μια hash συνάρτηση. Ας υποθέσουμε ότι το αρχικό γεγονός είναι η συμβολοσειρά ¨initial string¨. Ξεκινώντας από αυτό το γεγονός εφαρμόζουμε τη hash συνάρτηση και προκύπτει ένα αποτέλεσμα. Στη συνέχεια, εφαρμόζουμε σε αυτό το αποτέλεσμα τη hash συνάρτηση κοκ. Παρακάτω παρουσιάζουμε οπτικά αυτή τη διαδικασία:

Index	Εκτέλεση	Αποτέλεσμα
1	$hash(\text{"}initial\ string\text{"})$	$hash_1$
2	$hash(hash_1)$	$hash_2$
3	$hash(hash_2)$	$hash_3$
...	...	...
n	$hash(hash_{n-1})$	$hash_n$

Πίνακας 4.1: *Διαδοχική εφαρμογή της hash συνάρτησης*

Με βάση των ιδιοτήτων των hash συναρτήσεων μπορούμε με ασφάλεια να πούμε ότι το $hash_n$ θα ήταν διαφορετικό αν η αρχική τιμή ήταν διαφορετική. Επίσης, μπορούμε να υποθέσουμε με την ίδια σιγουριά ότι το $hash_{i-1}$ δημιουργήθηκε πριν από το $hash_i$. Έτσι, μπορούμε να συνδυάσουμε και άλλα γεγονότα με το κάθε hash και να τα εισάγουμε στη χρονική ακολουθία και έτσι να έχουμε μια χρονική διάταξη για το πότε αυτά συνέβησαν:

Index	Εκτέλεση	Αποτέλεσμα
1	$hash(\text{"}initial\ string\text{"})$	$hash_1$
2	$hash(comb(event_1, hash_1))$	$hash_2$
3	$hash(comb(event_2, hash_2))$	$hash_3$
...	...	...
n	$hash(comb(event_n, hash_{n-1}))$	$hash_n$

Πίνακας 4.2: *Διαδοχική εφαρμογή της hash συνάρτησης με δεδομένα*

Πλέον έχουμε ενσωματωμένα και τα γεγονοτα στην ακολουθία και γνωρίζουμε ότι το γεγονός $event_{i-1}$ συνέβη πριν το $event_i$.

Ονομάζουμε generator τη δομή που παράγει τις παραπάνω ακολουθίες. Μπορούμε να έχουμε παραπάνω από έναν generators που να λαμβάνουν ίδια ή διαφορετικά δεδομένα. Αυτοί μπορούν να συνδυαστούν αν περιοδικά ανταλλάσουν το τελευταίο hash τους.

Υποθέτουμε ότι τη χρονική στιγμή i οι δύο generators ανταλλάσουν τα hashes τους. Επομένως, το $hash^a_{i+1}$ θα εξαρτάται από τα $hash^a_i$ και $hash^b_i$, δηλαδή κάθε γεγονός μετά την

Index	Hash	Άλλο hash
$i-1$	$hash^a_{i-1}$	
i	$hash^a_i$	$hash^b_{i-1}$
$i+1$	$hash^a_{i+1}$	

Index	Hash	Άλλο hash
$i-1$	$hash^b_{i-1}$	
i	$hash^b_i$	$hash^a_{i-1}$
$i+1$	$hash^b_{i+1}$	

Πίνακας 4.3: *Συγχρονισμός generators*

i χρονική στιγμή θα εξαρτάται από τα γεγονότα και των δύο generators. Το ίδιο ισχύει και για τον δεύτερο generator.

Ο συγχρονισμός μεταξύ generators ικανοποιεί τη μεταβατική ιδιότητα. Αν έχουμε τρεις generators A, B και C, τότε αν ο B συγχρονιστεί με τους A και C, τότε ο A μπορεί να γνωρίζει τη σειρά των γεγονότων του σε σχέση με τον C μέσω του B.

Κακόβουλες επιθέσεις

Ένας κακόβουλος generator θα μπορούσε να κατασκευάσει το ιστορικό τοποθετώντας τα γεγονότα με διαφορετική σειρά από αυτή που τα έλαβε. Ένας τρόπος για να αποφευχθεί μια τέτοια επίθεση είναι οι χρήστες να συνδυάζουν το γεγονός που θέλουν να εισάγουν στην ακολουθία, μαζί με κάποιο προηγούμενο γεγονός από την υπάρχουσα ακολουθία το οποίο θεωρούν έγκυρο.

Με τον τρόπο αυτό, όταν ο χρήστης θα δημιουργήσει το $Event_{40}$ θα συμπεριλάβει το $hash^a_{30}$ και επομένως αν η ακολουθία του generator δεν το περιέχει τότε αυτό θα σημαίνει ότι το γεγονός αυτό δεν προκύπτει για μια έγκυρη ακολουθία.

Index	Hash	Γεγονότα
10	$hash^a_{10}$	
20	$hash^a_{20}$	$Event_{20}$
30	$hash^a_{30}$	$Event_{30}$
40	$hash^a_{40}$	$Event_{40}$

Index	Hash	Άλλο hash
10	$hash^b_{10}$	
20	$hash^b_{20}$	$Event_{40}$
30	$hash^b_{30}$	$Event_{30}$
40	$hash^b_{40}$	$Event_{20}$

Πίνακας 4.4: *Κακόβουλη επίθεση και αντιμετώπιση*

Σαν επιπλέον ασφάλεια, οι χρήστες μπορούν ακόμα να εισάγουν στο δίκτυο μια υπογραφή του παραπάνω hash.

4.2.1 Proof Of History στο Solana

Όλη η παραπάνω διαδικασία υλοποιείται στο Solana. Εδώ, υπάρχει ένας κόμβος του δικτύου που έχει το ρόλο του generator, δηλαδή λαμβάνει όλες τις τρέχουσες συναλλαγές και δημιουργεί το Proof Of History. Για κάθε συναλλαγή που λαμβάνει, την εκτελεί και αλλάζει την κατάσταση του δικτύου.

Ο generator λειτουργεί για ένα συγκεκριμένο χρονικό διάστημα και όταν αυτό το διάστημα λήξει, τότε γνωστοποιεί το ιστορικό στους υπόλοιπους κόμβους οι οποίοι έχουν το ρόλο του validator και ξανατρέχουν όλες τις συναλλαγές για να επιβεβαιώσουν ή οχι το αποτέλεσμα

του generator με θετική ψήφο. Στην περίπτωση που ο generator δεν επιβεβαιώνεται, τότε η απουσία ψήφου σημαίνει όχι.

Τα χρονικά διαστήματα για τα οποία κάποιος κόμβος είναι generator ονομάζονται slots, ενώ το κριτήριο για να επιλεγεί ένας κόμβος ως generator προκύπτει από το Proof of Stake το οποίο θα αναλυθεί στην επόμενη ενότητα.

Ταχύτητα επιβεβαίωσης

Ο generator παράγει σειριακά το ιστορικό για το Proof of History προκειμένου να ισχύει η χρονική αλληλουχία μεταξύ των γεγονότων. Ωστόσο, θα ήταν πολύ κοστοβόρο και θα επέφερε μεγάλη χρονική καθυστέρηση στο δίκτυο αν κάθε validator έπρεπε ελέγξει σειριακά τα γεγονότα του ιστορικού.

Για το λόγο αυτό, ο generator ανά συγκεκριμένα διαστήματα hashes γνωστοποιεί την κατάσταση του εκείνη τη στιγμή μαζί με το index της. Έτσι, τα διαστήματα χωρίζονται σε μικρότερα και εφόσον ο validator γνωρίζει τα αρχικά hashes μπορεί να παράξει μόνο εκείνα τα μικρά διαστήματα. Αυτό σημαίνει ότι με τη χρήση κατάλληλου hardware όπως GPU μπορεί να παραλληλοποιήσει τη διαδικασία και να πετύχει πολύ μικρότερους χρόνους επιβεβαίωσης για το ιστορικό.

Index	hash
100	$hash_{100}$
200	$hash_{200}$
300	$hash_{300}$
400	$hash_{400}$

Πίνακας 4.5: *Γνωστοποίηση κωδικών hash ανά 100 συναλλαγές*

Πλέον ο validator ελέγχει τα διαστήματα $[100, 200)$, $[200, 300)$ και $[300, 400)$ παράλληλα και παράγει την ψήφο του στο $\frac{1}{3}$ του χρόνου που χρειάστηκε ο generator.

4.3 Proof of Stake

Το Proof of Stake αποτελεί ένα πρωτόκολλο για την επίτευξη consensus μεταξύ των κόμβων του δικτύου. Το μεγαλύτερο κίνητρο για την χρήση του είναι το πολύ χαμηλότερο κόστος του σε ενέργεια σε σχέση με το παραδοσιακό Proof of Work. Ο λόγος που συμβαίνει αυτό θα γίνει αντιληπτός στη συνέχεια της ενότητας, όμως περιληπτικά μπορούμε να πούμε ότι πλέον η πιθανότητα να καταφέρει ένας κόμβος να κάνει mine ένα νέο block δεν εξαρτάται από την υπολογιστική του δύναμη αλλά από το ποσό του νομίσματος που διαθέτει.

Στο Proof of Stake consensus υπάρχει ένα σύνολο κόμβων που έχουν το ρόλο του miner. Οι κόμβοι αυτοί προσφέρουν ένα ποσό του νομίσματος με αντάλλαγμα να έχουν μια πιθανότητα να επιλεγούν για να είναι αυτοί οι επόμενοι που θα εισάγουν το νέο block στην αλυσίδα. Ο τρόπος που γίνεται η επιλογή για τον νέο κόμβο μπορεί να είναι είτε τυχαίος, είτε με βάση το ποσό που έχει αυτός διαθέσει. Στην περίπτωση που κάποιος κόμβος δουλέψει

κακόβουλα, τότε τιμωρείται από το δίκτυο και ένα ποσό από τα τοποθετημένα του νομίσματα αποσύρεται. Στο Proof of Stake, το ποσό που καταβάλει ο κόμβος ονομάζεται bond ενώ η διαδικασία τιμωρίας του και αφαίρεσης του ποσού ονομάζεται slashing.

Οι λόγοι για τους οποίους ένας miner μπορεί να δεχθεί slashing ποικίλουν. Οι πιο αθώοι είναι τεχνικοί, όπως μια αποτυχημένη σύνδεση στο διαδίκτυο ή ένα πρόβλημα του hardware. Η πιο κακόβουλη περίπτωση όμως είναι ο κόμβος να παράξει δύο block και να δημιουργήσει ένα τεχτητό fork στην αλυσίδα για να πετύχει double-spending και να ελπίζει ότι το κλαδί με την επίθεση θα επιβιώσει. Ωστόσο, τέτοιου είδους επιθέσεις είναι αναμενόμενες και ένα Proof of Stake σύστημα τις λαμβάνει υπόψιν του και τις ξεπερνά μέσω του slashing.

4.3.1 Proof of Stake στο Σολανα

Εδώ, το Proof of Stake δεν παίζει το ρόλο του mining, αλλά μέσω Proof of Stake μπορεί να επιλεγεί ο επόμενος generator για το Proof of History. Κάθε validator έχει μια πιθανότητα βάσει του stake του να επιλεγεί σαν generator. Οι validators παρακολουθούν τον generator και όσοι έχουν κάποιο βονδ μπορούν να ψηφίσουν πάνω στην νέα ακολουθία που αυτός παρήγαγε. Για να γίνει μια νέα ακολουθία αποδεκτή θα πρέπει τουλάχιστον τα $\frac{2}{3}$ των validators να ψηφίσουν θετικά.

Το bond που έχουν καταθέσει οι validators δεν εισάγεται στιγμαία στο δίκτυο. Όταν κάποιος validator επιθυμεί να προσθέσει ένα ποσό για staking, τότε αυτό μπαίνει σταδιακά. Το ίδιο ισχύει και για την αφαίρεσή του. Αυτό συμβαίνει για να αποφευχθούν οι κακόβουλες συμπεριφορές που θα συζητήσουμε στη συνέχεια.

Οι εκλογές στο Solana συμβαίνουν κάθε φορά που ένας generator αποτυγχάνει και αυτό μπορεί να γίνει για διάφορους λόγος. Πέρα από τον επόμενο generator, επιλέγεται και ένας αναπληρωματικός στην περίπτωση που ο πρώτος αποτύχει. Το σύστημα είναι έτσι σχεδιασμένο ώστε ένας generator να αποτυγχάνει τεχνητά μετά από κάποιο χρονικό διάστημα και να εναλλάσσεται με τον επόμενο.

Ένας κακόβουλος generator μπορεί να δημιουργήσει ένα τεχνητό fork στο δίκτυο δημιουργόντας δύο blocks με διαφορετικές συναλλαγές. Στην περίπτωση αυτή, αν βρεθεί κάποιος validator που ψηφίζει ταυτόχρονα για περισσότερα από ένα κλαδιά της αλυσίδας, τότε τιμωρείται με slashing. Ωστόσο υπάρχει και το πρόβλημα ότι οι validators θα είναι μοιρασμένοι σε περισσότερα κλαδιά άρα θα υπάρχουν προβλήματα στη συλλογή των κατάλληλων ψήφων για την επίτευξη consensus.

Η αποκατάσταση του δικτύου γίνεται σταδιακά με κριτήριο την συνέπεια έναντι της διαθεσιμότητας. Όταν δημιουργείται ένα fork, αν ένα κλαδί έχει περισσότερους από τα $\frac{2}{3}$ των validators, τότε πολύ σύντομα αυτοί θα μπορούν να αποδεσμευτούν και να μην μετράνε στις εκλογές για το consensus, έτσι ωστε να μπορέσουν να μεταβούν στη σωστή αλυσίδα. Αν υπάρχουν validators ανάμεσα στο $\frac{1}{2}$ και τα $\frac{2}{3}$ του δικτύου, τότε αυτοί θα αποδεσμευτούν σχετικά πιο αργά. Τέλος, σε κλαδί με λιγότερο από το $\frac{1}{2}$ των validators το consensus δεν θα μπορεί να επιτευχθεί σύντομα καθώς θα πρέπει να φύγει ένα πολύ μεγάλο μέρος των μη διαθέσιμων validators άρα το κλαδί αυτό θα είναι επί της ουσίας ανενεργό.

Έτσι, το Solana εξασφαλίζει ότι σε βάθος χρόνου το σωστό κλαδί, δηλαδή αυτό με το μεγαλύτερο πλήθος validators θα υπερισχύσει και το δίκτυο θα αποκαταστήσει την ακε-

ραιότητά του. Όλη αυτή η διαδικασία περιορίζει σημαντικά έναν κόμβο να λειτουργήσει κακόβουλα βάζοντας σε κίνδυνο το ίδιο το ποσό που έχει διαθέσει.

Staking δια αντιπροσώπου

Γενικά το Proof of Stake είναι ένα πρωτόκολλο που δεν απαιτεί πολλούς υπολογιστικούς πόρους. Σε αντίθεση με το Proof of Work, ένας συμμετέχων θα μπορούσε να χρησιμοποιεί ακόμα και ένα κινητό τηλέφωνο για να λάβει μέρος. Ωστόσο, η παραγωγή του Proof of History απαιτεί κάποιους πόρους και επειδή ο κάθε validator μπορεί εν δυνάμει να επιλεγεί για generator, δεν είναι εύκολο κάποιος να συμμετάσχει στο Proof of Stake.

Για να αντιμετωπιστεί αυτό το πρόβλημα, το Solana επιτρέπει μια μορφή staking κατά την οποία ένας λογαριασμός λειτουργεί σαν validator και generator χωρίς να έχει τους αντίστοιχους πόρους. Στην πραγματικότητα, ο χρήστης δημιουργεί έναν staking λογαριασμό στον οποίο μεταφέρει τα bonds που θέλει να δεσμεύσει για το staking. Στη συνέχεια, συνδέει τον λογαριασμό αυτό με κάποιον validator που εμπιστεύεται και το βονδ του validator αυξάνεται δίνοντας του μεγαλύτερη πιθανότητα να επιλεγεί ως generator. Ο χρήστης που έβαλε stake κερδίζει ένα ποσοστό από τα κέρδη του validator.

Ωστόσο, το γεγονός ότι τα bonds του χρήστη έχουν μεταφερθεί στον validator σημαίνει ότι ο χρήστης δεν μπορεί να τα χρησιμοποιήσει. Η διαδικασία της σύνδεσης αλλά και της αποσύνδεσης των bonds από το stake του validator είναι σχετικά αργή και γίνεται σταδιακά, ώστε κάποιος χρήστης να μην λειτουργήσει στιγμιαία κακόβουλα και βγει από το δίκτυο γρήγορα αλλά να τιμωρηθεί. Φυσικά, αν ο validator που έχει δεχθεί το bond λειτουργήσει εκτός κανονισμών, τότε το slashing που θα γίνει στο stake του θα επηρεάσει και τον λογαριασμό που τα κατέθεσε εκεί.

Άλλοι τρόποι επίθεσης στο δίκτυο

Θεωρητικά, ένας validator θα μπορούσε να μη σπαταλάει πόρους και απλά να επιβεβαιώνει κάθε ακολουθία που γνωστοποιεί ο generator. Αν παγιωθεί μια τέτοια κατάσταση, τότε ο κάθε generator θα μπορούσε να γνωστοποιεί ότι ακολουθία τον εξυπηρετεί με αποτέλεσμα το δίκτυο να χάσει την αξιοπιστία του. Έτσι, ο κάθε generator είναι ρυθμισμένος να δημιουργεί τεχνητά μια άκυρη συναλλαγή. Αν κάποιος validator την εγκρίνει, τότε τιμωρείται και γίνεται slashing επί του bond του.

Ένα άλλο είδος επίθεσης είναι το $\frac{1}{3}$ των validators να μην ψηφίσει το Proof of History του generator. Αυτό θα οδηγούσε σε πάγωμα του δικτύου καθώς δεν θα μπορούσε να υπάρξει η πλεοψηφία των $\frac{2}{3}$ για το consensus. Στην περίπτωση αυτή, το δίκτυο είναι σχεδιασμένο να δημιουργεί ένα τεχνητό fork στο οποίο θα συμμετέχουν μόνο οι αξιόπιστοι κόμβοι ενώ στους κακόβουλους θα απαγορευτεί η συμμετοχή. Σταδιακά, τα bonds τους θα αφαιρούνται έως ότου εν τέλει αφαιρεθεί το συνολικό ποσό.

4.4 Proof of Replication

Το Proof of Replication[6] είναι ένα κρυπτογραφικό πρωτόκολλο το οποίο χρησιμοποιείται για την επιβεβαίωση ότι ένας πάροχος πράγματι χρησιμοποιεί τον αποθηκευτικό χώρο

που ισχυρίζεται για να αποθηκεύσει δεδομένα. Αποτελεί ουσιαστικά μια επέκταση του Proof of Storage και στοχεύει στη δημιουργία αξιόπιστων κόμβων οι οποίοι έχουν αποθηκεύσει τα δεδομένα, μπορούν να τα ανακτήσουν κατά απαίτηση του χρήστη, ενώ μπορούν τέλος να επιβεβαιώνουν ανά πάσα στιγμή την κατοχή τους.

Τέτοια σχήματα είναι χρήσιμα για τις υπηρεσίες cloud, καθώς θέλουμε να είμαστε σίγουροι ότι τα δεδομένα μας υπάρχουν στο δίκτυο. Όσον αφορά τα αποκεντρωμένα δίκτυα, η σημασία τους είναι ακόμα μεγαλύτερη καθώς οι κόμβοι μπορεί να είναι κακόβουλοι και γενικότερα να μην υπάρχει εμπιστοσύνη μεταξύ τους. Όταν μιλάμε σε επίπεδο δημόσιων blockchains, είναι σημαντικό να εξασφαλίζεται η ύπαρξη και συντήρηση της πληροφορίας από τη στιγμή που οι κόμβοι είναι πρακτικά ανώνυμοι.

4.4.1 Χαρακτηριστικά

Για την εξασφάλιση των παραπάνω, ορίζουμε κάποιες απαιτήσεις για το σχήμα που θα χρησιμοποιήσουμε. Σαν ορολογία, ορίζουμε τον χρήστη που ζητά επιβεβαίωση ως V, τον πάροχο που καλείται να επιβεβαιώσει την κατοχή της πληροφορίας ως P, το άιτημα σε αυτόν ως c και την απόδειξη που αυτός παρέχει π^c. Στη συνέχεια, ορίζουμε τις εξής ιδιότητες που θέλουμε να έχει το σύστημα:

- **Ιδωτικά επιβεβαιώσιμο**: Ορίζουμε έτσι ένα σχήμα όταν ο χρήστης V κατέχει ένα ιδιωτικό κλειδί.
- **Δημόδια επιβεβαιώσιμο**: Ορίζουμε έτσι ένα σχήμα που ο χρήστης έχει πρόσβαση σε μια μορφή δημόσιας πληροφορία για τα δεδομένα, όπως για παράδειγμα κάποιο κλειδί επιβεβαίωσης, αλλά δεν έχει πρόσβαση στα αυθεντικά δεδομένα.
- **Διαφανές**: Ένα σχήμα στο οποίο ένας πάροχος P δεν μπορεί να δημιουργήσει κάποια έγκυρη απόδειξη π^c αν έχει πρόσβαση σε κάποια επιπλέον πληροφορία για τα δεδομένα.
- **Ανακτησιμότητα**: Ένα σχήμα ικανοποιεί το κριτήριο αυτό, όταν κάποιος χρήστης V μπορεί να ανακτήσει τα δεδομένα μέσα από διαδοχικά αιτήματα c προς τον πάροχο P.
- **Δυναμικό**: Δυναμικό είναι το σύστημα όταν επιτρέπει στον χρήστη να μεταβάλλει τα αποθηκευμένα δεδομένα.
- **Μη αναθέσιμο**: Όταν ο πάροχος P δεν μπορεί να αναθέσει την κατοχή των δεδομένων σε κάποιον τρίτο πάροχο P^*.
- **Επικυρώσιμο**: Είναι το σχήμα στο οποίο η ταυτότητα του παρόχου P μπορεί να επιβεβαιωθεί.
- **Χρονικά Περιορισμένο**: Είναι ένα σχήμα στο οποίο η εγκυρότητα μιας απόδειξης είναι περιορισμένη σε ένα ορισμένο χρονικό διάστημα.
- **Χρήσιμο**: Όταν ταυτόχρονα με την εκτέλεσή του μπορούν να πραγματοποιηθούν και άλλες χρήσιμες λειτουργίες.

Όλα τα παραπάνω συντελούν στη δημιουργία ενός αξιόπιστου συστήματος Proof of Storage. Αναλόγως το σκοπό μας, δεν είναι απαραίτητο να υλοποιούνται όλες οι απαιτήσεις.

4.4.2 Είδη Proofs

Το Proof of Storage έχει διάφορες μορφές διατηρώντας έναν κοινό σκοπό, την απόδειξη κατοχής δεδομένων. Συνεπώς, το Proof of Replication αποτελεί μια μορφή του Proof of Storage με συκεκριμένα χαρακτηριστικά. Για να γίνει καλύτερα κατανοητό, ορίζουμε τα εξής proofs:

- **Provable Data Possession (PDP)**: Τέτοια σχήματα επιτρέπουν στον χρήστη V να στέλνει δεδομένα D σε κάποιον πάροχο P και στη συνέχεια να μπορεί να επιβεβαιώσει ότι ο P πράγματι κατέχει τα δεδομένα.

- **Proof of Retrievability (PoRet)**: Είναι επέκταση του PDP, αλλά εδώ ο χρήστης μπορεί επιπλέον να κατασκευάσει τα δεδομένα D μέσα από τις αποδείξεις που του προσφέρει ο πάροχος P.

- **Proof of Storage (PoS)**: Είναι το γενικότερο σχήμα που επεκτείνει τα PDPs και PoRets.

- **Proof of Space (PoSpace)**: Επιτρέπει στον πάροχο P να αποδεικνύει ότι πράγματι έχει χρησιμοποιήσει κάποιο χώρο αποθήκευσης για τα δεδομένα.

- **Proof of Spacetime (PoSt)**: Εδώ ο πάροχος μπορεί να επιβεβαιώσει όχι μόνο ότι έχει χρησιμοποιήσει κάποιον αποθηκευτικό χώρο, αλλά ότι συνεχίζει να τον χρησιμοποιεί κατά το πέρας του χρόνου.

4.4.3 Επιθέσεις και απειλές

Το Proof of Replication αποτελεί έναν τύπο PoS ο οποίος εξασφαλίζει ότι ο πάροχος P έχει χρησιμοποιήσει φυσικό χώρο αποθήκευσης για τα ζητούμε δεδομένα. Υπάρχουν τρόποι οι πάροχοι P να εξαπατήσουν το σύστημα είτε με ψευδείς αποδείξεις ότι έχουν τα δεδομένα, είτε με πολλαπλές ταυτότητες προκειμένου να μεγιστοποιήσουν τα κέρδη τους.

Για παράδειγμα, κάποιος κακόβουλος πάροχος θα μπορούσε να αποθηκεύει τα δεδομένα μια φορά και να δημιουργήσει άλλες ταυτότητες που επικοινωνούν με τον χώρο αποθήκευσης και έτσι να φαίνεται ότι υπάρχουν περισσότεροι πάροχοι στο δίκτυο. Με τον τρόπο αυτό θα μπορούσε να πολλαπλασιάσει τα κέρδη του χωρίς να αφιερώσει πόρους.

Ένας άλλος τρόπος επίθεσης θα ήταν η ανάθεση της αποθήκευσης σε κάποιον τρίτο πάροχο και η ανάκτηση των δεδομένων μόνο όταν αυτά ζητηθούν από τον χρήστη του συστήματος. Αυτό δημιουργεί τον κίνδυνο ότι το σύστημα εξαρτάται στην πραγματικότητα από κάποιον άγνωστο πάροχο.

Τέλος, αν οι αποδείξεις για τα δεδομένα μπορούν με κάποιον τρόπο να παραχθούν γρήγορα κατά απαίτηση του συστήματος, ο πάροχος θα μπορούσε να μην αποθηκεύσει καν τα δεδομένα, αλλά μόνο τα απαραίτητα μέρη τους και έτσι να εξοικονομήσει χώρο αποθήκευσης παίρνοντας τα ίδια κέρδη.

4.4.4 Αντιμετώπιση επιθέσεων

Στην περίπτωση που θέλουμε να αποθηκεύσουμε πολλά αντίγραφα των δεδομένων σε κάποιον πάροχο, τότε μπορούμε να ορίσουμε για κάθε ξεχωριστό αντίγραφο κάποια συγκεκριμένη κωδικοποίηση. Με τον τρόπο αυτό, ακόμα και αν ο πάροχος έχει ψεύτικες ταυτότητες δεν θα μπορέσει να τις αξιοποιήσει καθώς θα πρέπει αναγκαστικά να έχει στην κατοχή του όλες τις κωδικοποιήσεις για να παρέχει αποδείξεις για όποια του ζητηθεί. Η κωδικοποίηση μπορεί να εξαρτάται από την ταυτότητα του παρόχου, επομένως η δυσκολία της εξαπάτησης αυξάνεται ακόμα περισσότερο καθώς και πάλι πολλές ταυτότητες εξαναγκάζουν σε πολλαπλές αποθηκεύσεις των δεδομένων.

Για να αποφύγουμε τα άλλα είδη επιθέσεων μπορούμε να ορίσουμε την προηγούμενη κωδικοποίηση με τέτοιον τρόπο έτσι ώστε η διαδικασία της κωδικοποίησης να είναι αρκετά πιο αργή από αυτή της κωδικοποίησης. Έτσι, ένας κακόβουλος πάροχος δεν θα επωφειλείται από την ανάθεση της αποθήκευσης σε κάποιον τρίτο καθώς τότε θα επιβαρύνεται και με τον χρόνο που απαιτείται για την απόκτηση των δεδομένων. Αν υποθέσουμε ότι ορίζουμε ένα χρονικό όριο για τη λήψη απάντησης περί της κατοχής των δεδομένων, τότε ο πάροχος θα θεωρηθεί άκυρος αφού θα καθυστερεί να απαντήσει.

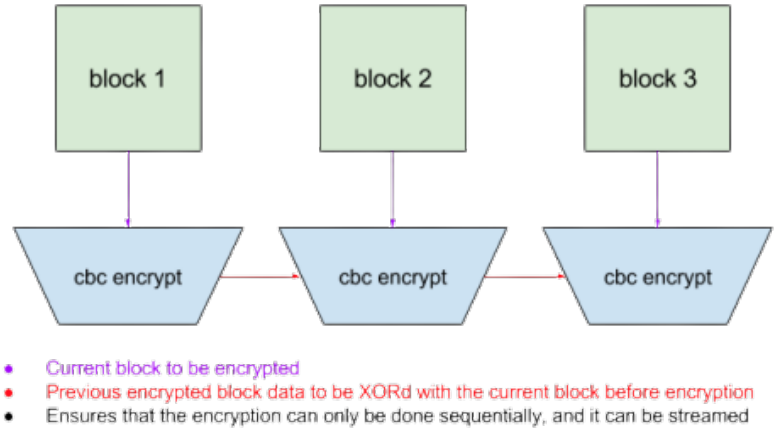
Φυσικά το ίδιο ισχύει και αν ο πάροχος σκοπεύει να παράξει τα δεδομένα εκείνη τη στιγμή καθώς θα επιβαρυνθεί με το κόστος όχι μόνο της παραγωγής αλλά και της κωδικοποίησης των δεδομένων.

4.4.5 CBC κωδικοποιήση

Για να επιτευχθούν τα παραπάνω ακολουθείται η εξής διαδικασία, η οποία ονομάζεται cipher block encryption. Χωρίζουμε τα δεδομένα που θα αποθηκευτούν σε τμήματα, τα οποία κωδικοποιούνται. Ωστόσο, για την κωδικοποίηση ενός τμήματος των δεδομένων απαιτείται ο συνδυασμός του με την κωδικοποίηση του προηγούμενου τμήματος. Έτσι, καταλήγουμε να έχουμε μια κωδικοποίηση η οποία εξαρτάται άμεσα και από την αλληλουχία των δεδομένων.

Επιπλέον, για να αποφύγουμε την περίπτωση κάποιος πάροχος να κωδικοποιεί τα δεδομένα μέχρι το σημείο όπου είναι απαραίτητο για να στείλει τα μηνύματα επιβεβαίωσης στον χρήστη, μπορούμε να κάνουμε μια αναδιάταξη στα blocks των δεδομένων. Έτσι, η παραγωγή του συνόλου των δεδομένων θα προϋποθέτει την κωδικοποίηση κάθε τμήματος ξεχωριστά. Το ίδιο θα ισχύει και για την αποκωδικοποίηση καθώς ο πάροχος δεν θα μπορεί να στείλει επιβεβαίωση για κάποιο τμήμα των δεδομένων αν πρώτα δεν έχει παράξει την συνολική κωδικοποιημένη εικόνα τους.

Τα παραπάνω δίνουν τη δυνατότητα να υπάρχει μια σειριακή κωδικοποίηση του συνόλου των δεδομένων με αποτέλεσμα κάποιος πάροχος να μην έχει τη δυνατότητα να παραλληλοποιήσει τη διαδικασία και να υποκλέψει το δίκτυο. Μπορούμε συμπληρωματικά να χρησιμοποιήσουμε κάποια μέθοδο κωδικοποίησης η οποία να είναι χρονικά παραμετροποιήσιμη ώστε να ρυθμίσουμε το χρόνο που θα απαιτείται από τον πάροχο για την κωδικοποίηση. Όσον αφορά την αντίστροφη διαδικασία είναι εύλογο να πούμε ότι αν ο πάροχος κατέχει συγκεκριμένη πληροφορία από την κωδιοκοποίηση, όπως για παράδειγμα τα αποτελέσματά της ανά block δεδομένων, τότε μπορεί να εκτελέσει παράλληλη αποκωδικοποίηση και να γνωστοποιήσει την απόδειξη σε πολύ μικρότερο χρόνο.



Εικόνα 4.1: *CBC κωδικοποίηση*

4.4.6 Proof of Replication στο Solana

Στο Solana υπάρχουν κόμβοι οι οποίοι αναλαμβάνουν να αποθηκεύουν όλα τα δεδομένα του συστήματος. Αυτοί οι κόμβοι έχουν σκοπό να εκτελούν μόνο αυτή τη διαδικασία καθώς αν το δίκτυο δουλεύει στους ρυθμούς που ορίζουν οι προδιαγραφές του, τότε υπολογίζεται ότι θα παράγει περίπου 4 petabytes δεδομένων ετησίως. Οι κόμβοι αυτοί ονομάζονται archivers [7].

Κάθε φορά που οι archivers έχουν κάποιο διαθέσιμο χώρο, τότε γνωστοποιούν στο δίκτυο αυτή τη διαθεσιμότητα. Ανά τακτά χρονικά διαστήματα, το δίκτυο χωρίζει το ιστορικό σε τμήματα και τα κατανέμει κατάλληλα στους archivers μέσω των validators. Αυτοί με τη σειρά τους θα πρέπει κάθε φορά που ερωτώνται για το αν έχουν αποθηκευμένα τα δεδομένα να παράγουν τις κατάλληλες αποδείξεις.

Κάθε archiver δημιουργεί ένα κλειδί υπογράφοντας ένα hash της ακολουθίας του Proof of History. Με τον τρόπο αυτό το κλειδί συνδέεται με την ταυτότητα του και με κάποια χρονική στιγμή στην αλυσίδα. Οι validators ανά τακτά χρονικά διαστήματα χρησιμοποιούν αυτά τα κλειδιά σαν σεεδ σε κάποιον ψευδοτυχαίο αλγόριθμο και παράγουν έναν αριθμό που χρησιομοποιείται σαν index για το κρυπτογραφημένο block. Η διαδικασία της παραγωγής του κλειδιού επαναλαμβάνεται περιοδικά ώστε οι archivers να μην μπορούν να προβλέψουν ποιό τμήμα των δεδομένων τους θα ζητηθεί.

Ένας archiver μπορεί σε τυχαίες στιγμές να παράξει κάποια ψευδή απόδειξη για τα δεδομένα. Οι validators είναι υποχρεωμένοι να εντοπίζουν τέτοιες περιπτώσεις. Στην περίπτωση που κάποιος validator δεν σημειώσει την απόδειξη σαν εσφαλμένη, τότε ο archiver μπορεί να το ανακοινώσει στο δίκτυο και το bond του validator να γίνει slash με τον αρχιερ να κερδίζει το 33% του βονδ του αλιδατορ. Για να αποφύγει τις κυρώσεις, ο αρχιερ ανακοινώνει στο δίκτυο ότι η απόδειξή του ήταν πράγματι εσφαλμένη και παρουσιάζει την συνάρτηση που χρησιμοποίησε για να την παράξει σαν αποδεικτικό.

Όλες οι παραπάνω επικοινωνίες καταγράφονται στο Proof of History και γίνονται μέρος του ιστορικού του δικτύου. Συνεπώς, μιλάμε για συναλλαγές και άρα όλες αυτές επιφέρουν

fees για τους validators και τους archivers. Ο λόγος που υπάρχουν fees είναι για να μην γίνεται αλόγιστη χρήση του δικτύου.

4.5 Άλλες καινοτομίες του Solana

Εκτός από τα παραπάνω, το Solana έχει εισάγει κι άλλες τεχνολογίες που αυξάνουν την ταχύτητά του. Αυτές παρουσιάζονται σε αυτή την ενότητα.

4.5.1 Tower BFT

Ο Tower BFT[8] αλγόριθμος αποτελεί μια παραλλαγή του PBFT και είναι σχεδιασμένος για να εκτελείται πάνω σε ένα δίκτυο με αντικειμενικότητα στο χρόνο. Το Proof of History του Solana έχει αυτή την αντικειμενικότητα καθώς ορίζει επακριβώς ποιές συναλλαγές και με ποιά σειρά έχουν εκτελεστεί από το δίκτυο.

Ο αλγόριθμος αυτός έχει σκοπό την επίτευξη consensus μεταξύ των κόμβων του δικτύου. Είναι σημαντικό να παρατηρήσουμε ότι εφόσον το Proof of History καταγράφει γεγονότα, τότε μπορεί να καταγράψει και τις κινήσεις των validators. Έτσι, η ύπαρξη μιας "τράπεζας" που τοποθετούνται όλα τα γεγονότα οδηγεί στην αποφυγή της peer-to-peer επικοινωνίας μεταξύ των κόμβων με συνέπεια να μειώνεται κατά πολύ η τάχυτητα που μπορεί να επιτευχθεί το consensus.

Ας υποθέσουμε ότι οι validators καλούνται να ψηφίσουν για την εγκυρότητα ενός block. Αφού ένας κόμβος το επιβεβαιώσει, τότε τοποθετεί την ψήφο του στο Proof of History. Μια ψήφος που μπαίνει στο δίκτυο έχει ισχύ 2 slots. Αν τα 2 slots περάσουν και το block δεν έχει επιβεβαιωθεί, τότε η ψήφος ακυρώνεται. Ωστόσο, κάθε φορά που ο validator ψηφίζει για κάποιο block, τότε όλες οι προηγούμενες ψήφοι του διπλασιάζουν το χρόνο που θέλουν για να απορριφθούν. Αυτό συνεχίζεται εώς ότου ένα block συγκεντρώσει τα $\frac{2}{3}$ της πλειοψηφίας του δικτύου.

Στην περίπτωση του forking, ένας validator μπορεί να αναζητήσει και να συγκρίνει από το ιστορικό πόσοι ψήφοι υπάρχουν για κάθε κλαδί και να επιλέξει αυτό με τους περισσότερους. Ένας validator ανταμοίβεται για τη συνεισφορά του για κάθε ψήφο του που έχει φτάσει 2^{32} slots χρόνο απόρριψης. Επομένως, το κίνητρο της ψήφου κατευθύνεται προς το κλαδί με τη μεγαλύτερη πλειοψηφία.

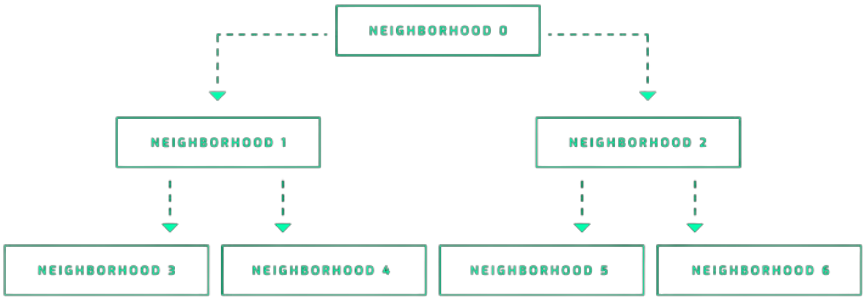
Όλα τα παραπάνω καταγράφονται στο ιστορικό ταυτόχρονα με τις συναλλαγές του δικτύου επομένως το Solana μπορεί και συνεχίζει τις συναλλαγές χωρίς να επηρεάζεται από τις ψηφοφορίες και τις καθυστερήσεις που μπορεί να επέλθουν από τον κλασσικό PBFT.

4.5.2 Turbine

Το Turbine [9] είναι το πρωτόκολλο μέσω του οποίου οι κόμβοι του Solana επικοινωνούν μεταξύ τους και μοιράζονται την πληροφορία. Σκοπός του είναι η ελαχιστοποίηση του χρόνου που χρειάζεται για να γνωστοποιηθεί η πληροφορία ενός block σε όλο το δίκτυο.

Στην περίπτωσή μας, οι κόμβοι του δικτύου οργανώνονται σε γειτονιές. Η κάθε γειτονιά μπορεί να περιέχει πολλούς κόμβους, ενώ η σύνδεση μεταξύ των γειτονιών δημιουργεί μια δενδρική δομή. Έτσι, ο generator μοιράζει το περιεχόμενο του σε πολλά πακέτα στην αμέσως

επόμενη γειτονιά από αυτόν. Κάθε κόμβος της γειτονιάς είναι υπεύθυνος να μεταφέρει τα πακέτα που λαμβάνει σε όλους τους υπόλοιπους κόμβους της γειτονιάς αλλά και ένα μέρος τους στους κόμβους της κατώτερης γειτονιάς κοκ. Έτσι, αν υποθέσουμε ότι κάθε γειτονιά αποτελείται από 200 κόμβους και έχουμε ένα δίκτυο τριών επιπέδων, τότε ένα μήνυμα μπορεί να φτάσει μέχρι και σε 40000 κόμβους με δύο αποστολές.



Σχήμα 4.2: *Γειτονιές κόμβων*

Οι κόμβοι στις γειτονιές οργανώνονται με βάση το stake τους έτσι ώστε να αποφευφχθεί η κακόβουλη δραστηριότητα καθώς ένας κόμβος με μεγάλο stake έχει περισσότερα να χάσει παρά να κερδίσει αν δεν λειτουργήσει σύμφωνα με τους κανόνες. Μια κακόβουλη επίθεση μπορεί να σημαίνει μη αποστολή μηνυμάτων σε κατώτερα επίπεδα ή παραποίηση των μεταδιδόμενων μηνυμάτων. Για το λόγο αυτό τα μηνύματα είναι κωδικοποιημένα σε Reed-Solomon έτσι ώστε ακόμα και αν χαθεί πληροφορία, αυτή να μπορεί να ανακτηθεί από τα πακέτα που λήφθηκαν επιτυχώς.

4.5.3 Gulf Stream

Το Gulf Stream[10] είναι το σύστημα με το οποίο το Solana διαχειρίζεται το mempool των συναλλαγών του. Σαν mempool ορίζουμε τη δομή που διατηρεί όλες τις μη εκτελεσμένες συναλλαγές. Κάθε κόμβος διατηρεί το δικό του mempool. Είναι λογικό να ισχυριστούμε ότι όσο μεγαλύτερο είναι το mempool, τόσο πιο αργό γίνεται ένα blockchain καθώς σπαταλάει περισσότερους πόρους για τη διαχείρισή του.

Στο Solana το mempool λειτουργεί με διαφορετικό τρόπο από τα υπόλοιπα blockchains. Αρχικά, όταν ο χρήστης στέλνει μια συναλλαγή στο δίκτυο είναι υποχρεωμένος να στείλει και το hash ενός πρόσφατου block. Αυτό συμβαίνει για 2 λόγους. Πρώτον, η ύπαρξη ενός πρόσφατου blockhash παρέχει την ασφάλεια ότι ο χρήστης παρακολουθεί το σωστό ιστορικό. Δεύτερον, το blockhash ορίζει την ημερομηνία λήξης της συναλλαγής ως εξής: Αν μια συναλλαγή γίνει παλαιότερη από 32 blocks, τότε θεωρείται άκυρη. Έτσι, το mempool μπορεί και ανανεώνεται συνεχώς και στην περίπτωση που σε μια χρονική στιγμή υπήρξε μια απότομη αύξηση των συναλλαγών που δεν ήταν δυνατόν να τη διαχειριστεί, τότε θα ξεφύγει από αυτή την κατάσταση σε 32 blocks.

Επιπλέον, όταν οι κόμβοι κάνουν εκλογές δεν αποφασίζουν μόνο για τον επόμενο generator αλλά για αρκετούς μελλοντικούς. Επομένως, αντί να γνωστοποιούν τις συναλλαγές που

λαμβάνουν μόνο στον τρέχων generator, τις γνωστοποιούν και σε άλλους μελλοντικούς και αυτοί με τη σειρά του εκτελούν όσες είναι δυνατόν να εκτελεστούν νωρίτερα. Τέλος, η επιτάχυνση προκύπτει και από το γεγονός ότι αντί να πρέπει όλοι οι κόμβοι να επικοινωνήσουν μεταξύ τους για να γνωστοποιοήσουν τις τρέχουσες συναλλαγές, υπάρχουν συγκεκριμένοι κόμβοι παραλήπτες και σαν συνέπεια η επικοινωνία μεταξύ τους ελαχιστοποιείται.

4.5.4 Sealevel

Το Sealevel[11] είναι το runtime του Solana, όπου επιτρέπει ταχύτατη εκτέλεση των προγραμμάτων του. Για να το πετύχει αυτό, το Solana έχει δημιουργήσει τις κατάλληλες προϋποθέσεις μέσω της δομής των συναλλαγών του. Επιπλέον, είναι σχεδιασμένο ώστε να χρησιμοποιεί CUDA και να εκμεταλλεύεται την παραλληλοποίηση που μπορεί να προσφέρει το hardware που χρησιμοποιεί.

Όπως είδαμε στην πρώτη ενότητα, όταν ο χρήστης κατασκευάζει μια συναλλαγή, τότε θα πρέπει να συμπεριλάβει σε αυτή και τους λογαριασμούς από τους οποίους αυτή θα διαβάσει και θα γράψει. Αυτό μας δίνει τη δυνατότητα να διαχωρίσουμε τους λογαριασμούς βάσει των εξαρτήσεων που έχουν μεταξύ τους οι εντολές που θέλουμε να εκτελλέσουμε και να εκτελέσουμε στη συνέχεια τα ανεξάρτητα κομμάτια παράλληλα.

Περιληπτικά, ένας validator θα οργανώσει όλες τις συναλλαγές που έχει στο mempool του και θα συγκεντρώσει τα δεδομένα όλων των λογαριασμών που χρησιμοποιούνται. Στη συνέχεια, θα ομαδοποιήσει αυτούς τους λογαριασμούς μαζί με τα προγράμματα που τους χρησιμοποιούν και σύμφωνα με τις εξαρτήσεις που έχουν μεταξύ τους και ύστερα θα εκτελέσει κάθε ανεξάρτητη ομάδα παράλληλα. Η τεράστια επιτάχνυση έρχεται επίσης όταν πρόκειτα να χρησιμοποιήσουμε προγράμματα του συστήματος όπως το Token Program. Σε αυτή την περίπτωση, μια GPU μπορεί να εκτελέσει ταυτόχρονα χιλιάδες συναλλαγές για το πρόγραμμα αυτό καθώς ακολουθεί το μοντέλο SIMD (Single Instruction Multiple Data). Δεδομένου ότι η πλειοψηφία των συναλλαγών στο δίκτυο περιλαμβάνουν κλήσεις στα συστημικά προγράμματα, η επιτάχυνση που μας δίνουν οι GPU είναι τεράστια.

Όλα τα παραπάνω έρχονται σε αντίθεση με την single-threaded εκτέλεση των smart contracts σε blockchain όπως το Ethereum και δικαιολογούν την ταχύτητα με την οποία λειτουργεί το Solana.

4.5.5 Pipelining

Το pipelining[12] αφορά την εκτέλεση των συναλλαγών από τον generator και την επαλήθευσή τους από τους validators. Για το σκοπό αυτό, το Solana υλοποιεί προπγραμματιστικά τη λογική του pipeline που χρησιμοποιεί και ένας επεξεργαστης σε επίπεδο hardware. Συγκεκριμένα, ένας κόμβος έχει δύο επίπεδα λειτουργίας, το TPU (Transaction Processing Unit) που αφορά τους generators και το TVU (Transaction Validator Unit). Και τα δύο επίπεδα λειτουργίας απαρτίζονται από τα ίδια στάδια τα οποία είναι τα εξής:

- Fetch (Kernel Space)
- Signature Verify (GPU)

- Banking(CPU)
- Write (Kernel Space)

Παραπάνω συμπεριλαμβάνουμε και τις μονάδες που εκτελούν την κάθε λειτουργία. Όπως μπορεί να συμπεράνει κανείς και από το όνομα της υποενότητας, κάθε στιγμή το Solana εκμεταλλεύεται και τα 4 επίπεδα λειτουργίας τα οποία εκτελούν ταυτόχρονα λειτουργίες. Η επιβεβαίωση των συναλλαγών γίνεται στην GPU καθώς είναι η πιο κοστοβόρα αλλά ταυτόχρονα και η πιο παραλληλοποιήσημη διαδικασία.

4.5.6 Cloudbreak

Το Cloudbreak[13] είναι ένα σύστημα που κατασκεύασε η ομάδα του Solana με σκοπό να έχει πολύ γρήγορη πρόσβαση στους λογαριασμούς των χρηστών. Η κύρια διαφορά σε σχέση με τη leveldb που χρησιμοποιούν άλλα blockchains για να αποθηκεύουν και να προσπελαύνουν τους λογαριασμούς είναι ότι πλεόν μπορεί να γίνεται παράλληλη προσπέλαση. Η τεχνολογία αυτή εκμεταλλεύεται στο μέγιστο τις ικανότητες των SSD δίσκων να υποτηρίζουν πολλά ταυτόχρονα threads και να επιτυγχάνουν μέχρι και 370000 αναγνώσεις το δευτερόλεπτο.

# Accounts	Fork	Total writes	Date size	Read threads	Write threads	Writes	Reads*
1,000,000	10	10,000,000	0	1	1	1,088,257.80	1,182,838.60
1,000,000	10	10,000,000	512	1	1	625,312.70	960,211.80
1,000,000	10	10,000,000	1024	1	1	469,417.44	829,627.06
1,000,000	10	10,000,000	2048	1	1	293,349.75	740,489.75
1,000,000	10	10,000,000	2048	1	1	293,349.75	740,489.75
1,000,000	10	10,000,000	0	16	16	1,380,834.00	13,962,485.00
1,000,000	10	10,000,000	1024	16	16	954,836.25	9,563,040.00
10,000,000	10	100,000,000	1024	16	16	701,852.00	1,604,101.50

Πίνακας 4.6: *Benchmarking του Cloudbreak*

Αντί να χρησιμοποιείται κάποια βάση δεδομένως σαν διεπαφή για την πρόσβαση στους λογαριασμούς, εδώ χρησιμοποιούμε memory-mapped αρχεία που επί της ουσίας είναι αρχεία που το περιεχόμενό τους είναι αντιστοιχισμένο στον εικονικό χώρο διευθύνσεων μιας διεργασίας. Πλέον το bottleneck είναι ο χρόνος προσπέλασης της μνήμης που εξαρτάται από την τεχνλογία των δίσκων, αλλά είναι αρκετά μικρός.

Κατά τη λειτουργία ενός κόμβου το ευρετήριο των λογαριασμών βρίσκεται στη RAM. Επίσης, η πρόσβαση και η ανανέωση των λογαριασμών γίνεται copy-on-write, επομένως αυτό μας δίνει μεγάλη επιτάχυνση ειδικά όταν δεν θέλουμε να αλλάξουμε το περιεχόμενο κάποιου λογαριασμού. Συνδυάζοντας τα παραπάνω, το Solana πετυχαίνει πολύ γρήγορες αναζητήσεις λογαριασμών το οποίο έχει σαν επακόλουθο την πολύ γρήγορη εκτέλεση των συναλλαγών.

Ο παραπάνω πίνακας μας παρουσιάζει ανά δευτερόλεπτο τις ενέργειες που πετυχαίνει το Cloudbreak. Βλέπουμε ότι για 10 εκατομμύρια λογαριασμούς μπορεί να πετύζει μέχρι και 700000 περίπου εγγραφές γεγονός που αποδεικνύει το πώς οι συαλλαγές στο Solana εκτελούνται τόσο γρήγορα.

Μέρος II

Πειραματικό Μέρος

Κεφάλαιο 5

Σύγκριση

Στο κεφάλαιο αυτό θα κάνουμε μια συγκριτική αποτίμηση μεταξύ των blockchains Ethereum και Solana. Ο λόγος που επιλέχθηκε το πρώτο αφορά τη δημοτικότητά του, καθώς σήμερα αποτελεί την κύρια πλατφόρμα διεκπεραίωσης συναλλαγών και smart contracts στο χώρο των blockchains. Το δεύτερο επιλέχθηκε έτσι ώστε να αντιληφθούμε καλύτερα τον τρόπο με τον οποίο οι σχεδιαστικές επιλογές και οι βελτιστοποιήσεις που έχει εισάγει οδηγούν σε ένα blockchain με επιδόσεις αντίστοιχες ενός μη αποκεντρωμένου συστήματος.

Παρά τις διαφορές που θα δούμε, αξίζει να επισημάνουμε ότι κάθε blockchain γενικότερα αποσκοπεί στη λύση ενός προβλήματος. Κατά συνέπεια, δεν υπάρχει κάποιο το οποίο θα μπορούσε να χρησιμοποιηθεί μεμονωμένα. Το Bitcoin άνοιξε το δρόμο για την ευρεία αποδοχή της τεχνολογίας αυτής ενώ ταυτόχρονα ικανοποίησε την ανάγκη για ασφαλείς και ταυτόχρονα ανώνυμες συναλλαγές μεταξύ μερών. Το Ethereum με τη σειρά του δημιουργήθηκε με σκοπό να επεκτείνει τη λειτουργικότητα του Bitcoin και να επιτρέψει στους χρήστες να εκτελούν πιο περίπλοκες συναλλαγματικές διαδικασίες μέσω των smart contracts. Τέλος, το Solana ήταν μια πρακτική εφαρμογή πρωτοκόλλων consensus και αξιοπιστίας η οποία οδήγησε σε εκπληκτικά αποτελέσματα, ωστόσο ακόμα η αξιοπιστία κλονίζεται από αδυναμίες του να ανταπεξέλθει σε μεγάλο φόρτο εργασίας.

5.1 Μεθοδολογία σύγκρισης

Τα παραπάνω blockchains θα συγκριθούν σε 4 επίπεδα με σκοπό να αποκτήσουμε μια συνολική εικόνα των δυνατοτήτων τους. Από τη στιγμή που ένα τέτοιο σύστημα αποτελείται από πολλά επιμέρους τμήματα δεν είναι δυνατόν να το εξετάσουμε αποκλειστικά μακροσκοπικά και να μετρήσουμε για παράδειγμα μόνο τον χρόνο απόκρισης για τις συναλλαγές. Αν προχωρούσαμε σε μια τέτοια καταγραφή θα είχαμε ελλειπή εικόνα.

Ένα σύστημα μπορεί να είναι μεν αργό σε κάποιο επίπεδο αλλά οι δυνατότητες που έχει σε κάποιο άλλο θα μπορούσαν να το καταστήσουν χρήσιμο για κάποιο άλλο σκοπό. Για παράδειγμα, η βιβλιογραφία υπαγορεύει ότι το Solana προσφέρει πολύ μεγάλη ταχύτητα εκτέλεσης των smart contracts, ωστόσο ο κώδικάς του είναι πολύ περιοριστικός όσον αφορά το μέγιστο πλήθος υπολογισμών που μπορούμε να εκτελέσουμε. Αντιθέτως, το Ethereum αφήνει ένα πιο ελαστικό όριο στη δέσμευση των πόρων, επομένως θα μπορούσε να είναι μια πιο καλή επιλογή για εφαρμογές που απαιτούν αρκετή υπολογιστική ισχύ.

Τα επίπεδα της σύγκρισης περιορίζονται στα παρακάτω:

Εφαρμογής
Σε αυτό το επίπεδο θα εξετάσουμε τις επιδόσεις του δικτύου στην εκτέλεση μιας εφαρμογής. Αυτή η εφαρμογή θα περιλαμβάνει αναγνώσεις και εγγραφές από τη μνήμη, ενώ ταυτόχρονα θα εκτελεί κάποιους υπολογισμούς. Έτσι, θα μπορούμε να προσομοιώσουμε την εκτέλεση μιας πραγματικής εφαρμογής υπό τον φόρτο πολλών συναλλαγών και θα δούμε πώς το κάθε δίκτυο ανταπεξέρχεται σε αυτόν.

Εκτέλεσης
Αφορά μια εφαρμογή η οποία διαβάζει και γράφει σε κάποιες συγκεκριμένες δομές της μνήμης του προγράμματος, ωστόσο το κύριο κομμάτι της εκτέλεσής του αφορά υπολογιστικά βαριές πράξεις. Εδώ εξετάζουμε πώς ανταποκρίνεται το δίκτυο αποκλειστικά στις υπολογιστικές απαιτήσεις μιας εφαρμογής.

Δεδομένων
Πρόκειται για απλές προσπελάσεις μνήμης οι οποίες ανήκουν στο πρόγραμμα. Εδώ εξετάζουμε το χρόνο που χρειάζεται το δίκτυο για τη διαχείριση της μνήμης του.

Consensus
Εξετάζομαι καθαρά την ταχύτητα με την οποία το δίκτυο διεκπεραιώνει το consensus επίπεδο. Αυτό θα μπορούσε να πει κανείς ότι είναι τα πιο ενδιαφέρονται σημεία της εργασίας καθώς έχουμε σύγκριση μεταξύ τελείως διαφορετικών πρωτοκόλλων, ενώ το ένα μάλιστα αποτελεί καινοτομία του Solana.

5.1.1 Μετρικές

Για να αντληθούν τα απαραίτητα συμπεράσματα θα συγκεντρώσουμε τις παρακάτω μετρικές ώστε να έχουμε μια ολιστική σύγκριση μεταξύ των δύο συστημάτων:

Throughput
Σαν throughput ορίζουμε το πλήθος των συναλλαγών που είναι επιτυχής ανά μια μονάδα χρόνου. Στα πειράματά μας θα προσπαθήσουμε να φτάσουμε το σύστημα στα όρια του εξαντλώντας το throughput.

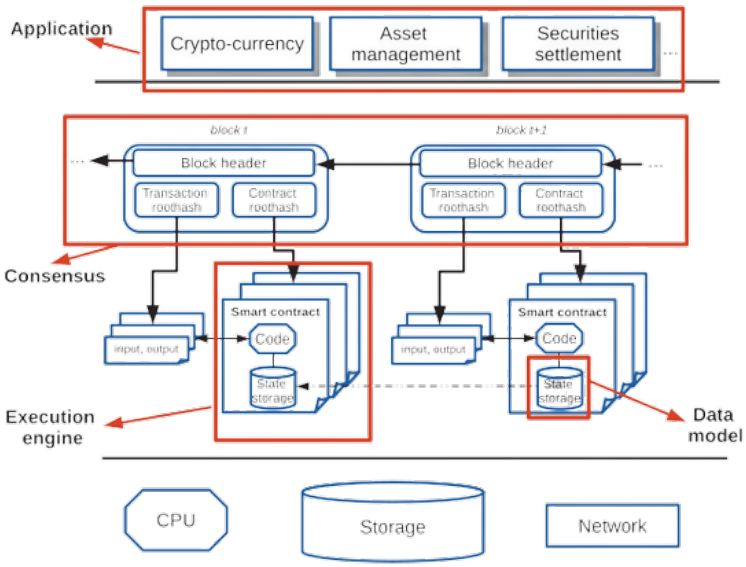
Latency
Είναι η ταχύτητα απόκρισης του συστήματος ανά συναλλαγή, δηλαδή ο χρόνος που χρειάζεται για να εκτελεστεί αυτή.

Scalability
Ως scalability ή επεκτασιμότητα ενός blockchain ορίζουμε τη δυνατότητά του να συνεχίζει να λειτουργεί βάσει προδιαγραφών ενώ αυξάνονται οι συμμετέχοντες κόμβοι.

5.2 Υλοποίηση πειραμάτων

Για τα πειράματα που εκτελέσαμε χρησιμοποιήσαμε ένα benchmark το οποίο ονομάζεται blockbench[14]. Αυτό περιέχει τον κατάλληλο κώδικα έτσι ώστε να δημιουργεί το κάθε δίκτυο και να αλληλεπιδρά με αυτό. Επίσης, έχει κατάλληλα σημεία κώδικα που μπορεί να τροποποιεί το δίκτυο και να σταματάει κόμβους έτσι ώστε να εξεταστεί η συμπεριφορά του σε τέτοιες περιπτώσεις.

Το benchmark αυτό δημιουργεί έναν φόρτο εργασίας ανάλογα με τις προδιαγραφές που έχουμε ορίσει και στέλνει τις κατάλληλες συναλλαγές στο δίκτυο



Εικόνα 5.1: *Τα επίπεδα στο blockbench*

έτσι ώστε να ανταποκρίνονται στο φόρτο αυτό. Στη συνέχεια, παρακολουθεί το δίκτυο και συλλέγει τις μετρήσεις.

Όσον αφορά τη γλώσσα στην οποία είναι υλοποιημένο το benchmark για το Ethereum, αυτή είναι κατά βάση η C++, όμως έχει και τμήματα γραμμένα σε Javascript με την βιβλιοθήκη web3.js. Σε αυτά τα τμήματα χρειάστηκε να γίνουν κάποιες προσαρμογές ώστε να ανταποκρίνονται στις ανανεώσεις που έχουν γίνει στο API του Ethereum από τότε που γράφτηκε το benchmark.

Το κομμάτι για το Solana συμπληρώθηκε από εμάς και είναι γραμμένο σε Rust. Ο λόγος είναι ότι η Rust έχει επιδόσεις παρόμοιες επιδόσεις με τη C++ ενώ ταυτόχρονα παρέχει ένα σύνολο πακέτων που κάνουν τη διεπαφή με το Solana πολύ εύκολη. Υπάρχουν επίσης κομμάτια γραμμένα σε Javascript με τη βιβλιοθήκη @solana/web3.js καθώς εκείνα τα κομμάτια δεν χρησιμοποιούν τον όγκο που παράγει το benchmark και μπορούν να εξεταστούν ανεξάρτητα.

5.2.1 Προγραμματιστική υλοποίηση των συμβολαίων

Για να υπολογίσουμε τις προηγούμενες μετρικές που αναφέραμε ανάλογα με το τμήμα του κάθε blockchain έπρεπε να κατασκευάσουμε τα αντίστοιχα smart contracts σε κάθε

αλυσίδα. Τα contracts του Ethereum παρέχονται από το benchmark, ενώ τα αντίστοιχα για το Solana έπρεπε να υλοποιηθούν ξανά.

Αυτά είναι τα εξής:

Smallbank

Εξετάζει το blockchain σε επίπεδο εφαρμογής. Σε αυτό το συμβόλαιο δεσμεύουμε σταδιακά μνήμη για ένα μια key-value μνήμη, την προσπελαύνουμε και εκτελούμε απλές αριθμητικές διαδικασίες.

CPU-Heavy

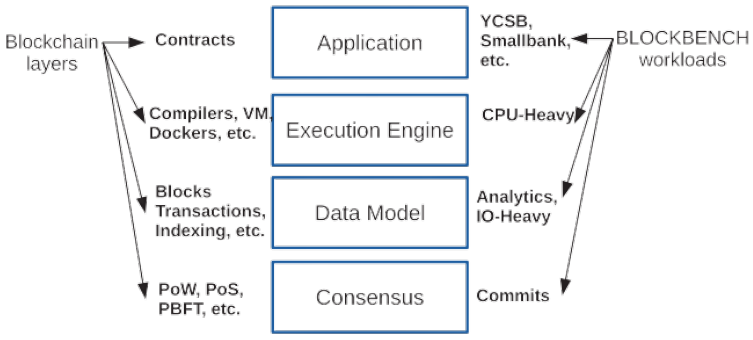
Εκτελεί πολύ βαριές πράξεις πάνω σε μεταβλητές του προγράμματος χωρίς να δεσμεύει κάποια παραπάνω μνήμη. Έτσι, μπορούμε να μελετήσουμε την απόδοση όσον αφορά το εκτελεστικό κομμάτι. Στην πράξη πρόκειτα για ένα contract το οποίο εκτελεί μια quicksort πάνω σε έναν εσωτερικό πίνακα.

IO-Heavy

Απλή προσπέλαση και ανανέωση τιμών μιας key-value μνήμης του προγράμματος. Εδώ δεν εκτελείται καμία παραπάνω πράξη και εξετάζουμε την ταχύτητα αλληλεπίδρασης με τη μνήμη του συστήματος.

Commits ή Do-Nothing

Είναι το συμβόλαιο που εξετάζει το consensus επίπεδο. Το περιεχόμενό του είναι κενό, εδώ το πρόγραμμα δεν περιέχει καθόλου κώδικα. Με τον τρόπο αυτό μπορούμε να δούμε την ταχύτητα με την οποία έρχεται το consensus χωρίς αυτή να επηρεάζεται από κάποιον άλλον παράγοντα.



Εικόνα 5.2: *Τα επίπεδα που αφορούν το κάθε contract*

Παραδοχές για τα πειράματα

Το Solana αποτελεί ένα δίκτυο με τελείως διαφορετική αρχιτεκτονική από το Ethereum. Επομένως, τα συμβόλαια σε αυτό πρέπει να γραφτούν με μια διαφορετική λογική, όπως και έγινε. Για παράδειγμα, το Solana έχει τον περιορισμό ότι ένας λογαριασμός μπορεί να έχει μέχρι 10MB χώρο αποθήκευσης. Καταλαβαίνουμε ότι κάτι τέτοιο είναι πολύ περιριστικό σε ένα benchmark που μπορεί να κάνει πολλές χιλιάδες καταχωρήσεις σε μια key-value δομή.

Ethereum	Solana
CPU με 4+ cores 16+ GB RAM 1TB SSD 25MBit/s	CPU με 12cores / 24threads + 128GB+ RAM 3 disks 500GB+/disk 300Mbit/s symmetric

Πίνακας 5.1: *Απαιτήσεις των blockchain*

Το προγραμματιστικό μοντέλο του Solana υποδεικνύει τη χρήση των PDAs, που αναφερθήκαμε στο προηγούμενο κεφάλαιο της εργασίας. Συνοπτικά, τα PDAs είναι διευθύνσεις λογαριασμών στις οποίες δεν αντιστοιχεί κάποιο ιδιωτικό κλειδί και μπορούν να χρησιμοποιηθούν χωρίς υπογραφή μέσα από το πρόγραμμα που τις κατέχει. Τα PDAs μπορούν να παραχθούν με τη χρήση αυθαίρετων seeds, ενώ είναι υποχρεωτική η εισαγωγή της διεύθυνσης του προγράμματος που τις παράγει. Επομένως, μπορούμε να αντικαταστήσουμε τη hashmap δομή με μια δομή *account* → *value*, όπου δηλαδή δημιουργούμε έναν νέο λογαριασμό για κάθε νέο κλειδί που θέλουμε να εισάγουμε, αναθέτουμε την κατάλληλη μνήμη και στη συνέχεια μπορούμε να χρησιμοποιούμε το λογαριασμό αυτό μέσα από το contract σαν μια κλασσική key-value δομή.

5.3 Πειράματα

5.3.1 Περιβάλλον πειραμάτων

Για την εκτέλεση των πειραμάτων χρησιμοποιήσαμε 14 εικονικές μηχανές του εργαστηρίου του CSLab, οι οποίες είχαν 8 πυρήνες, 16GB μνήμη RAM και 40 GB αποθηκευτικό χώρο. Αξίζει να εξετάσουμε ποιές είναι οι ελάχιστες προδιαγραφές που προτείνουν οι επίσημες σελίδες των blockchains που θα εξετάσουμε, έτσι ώστε να έχουμε μια άποψη για την ποιότητα των αποτελεσμάτων που αναμένουμε:

Όπως φαίνεται από τον πίνακα, και τα δύο συστήματα έχουν πολύ μεγαλύτερες απαιτήσεις από τις υπάρχουσες με το Solana να βρίσκεται σε πολύ πιο μακρινή θέση από το Ethereum. Επομένως, αναμένουμε ότι οι αποδόσεις και των δύο blockchains δεν θα ταιριάζουν με τις περιγραφές τους, ωστόσο μπορούμε να εκμεταλλευτούμε την κατάσταση αυτή και να δούμε πώς λειτουργούν τα blockchains αυτά με περιορισμένους πόρους.

Προετοιμασία

Η διαδικασία του μινινγ στο Ethereum επηρεάζεται πολύ από τον βαθμό δυσκολίας του συστήματος, ενώ για να προσαρμοστεί το difficulty χρειάζεται να περάσει ένα πολύ μεγάλο πλήθος από blocks στην αλυσίδα. Η σταθεροποίηση του συστήματος μπορεί να χρειαστεί ακόμη και μέρες και δεδομένου ότι τα πειράματα που έχουμε να εκτελέσουμε είναι πολλά, έγινε η επιλογή να εξετάζουμε συστηματικά το σύστημα έτσι ώστε το κάθε πείραμα να εκτελεστεί σε όσο το δυνατόν πιό σταθερό περιβάλλον. Ωστόσο, πρέπει να έχουμε κατά νου ότι το mining δεν είναι μια ντετερμινιστική διαδικασία όσον αφορά τον χρόνο περάτωσης και αυτό συμβαίνει ακόμα και στο πραγματικό δίκτυο.

Ο μόνος τρόπος που μπορούμε να επιταχύνουμε τη σταθεροποίηση του δικτύου στο Ετηερευμ είναι να θέσουμε μια σχετικά υψηλή τιμή στο difficulty από το genesis block έτσι ώστε οι αποκλίσεις του mining να κυμαίνονται γύρω από τα 15 δευτερόλεπτα, όπου είναι και ο χρόνος στόχος για το Ethereum.

Όσον αφορά το Solana, θεωρούμε ότι το σύστημα είναι έτοιμο όταν όλοι οι validators είναι εν δυνάμει leaders έτσι ώστε να εξασφαλίσουμε την κυκλική αποστολή του Proof of History. Στο Solana μπορούμε να το πετύχουμε αυτό σχετικά εύκολα μοιράζοντας ένα αρχικό ίσο stake σε όλους τους validators. Στη συνέχεια πρέπει να περιμένουμε το πέρασμα μερικών epochs ώστε το stake όλων να απορροφηθεί και να είναι έτοιμοι να γίνουν leaders. Η διαδικασία αυτή δεν χρειάζεται περισσότερο από 10 λεπτά.

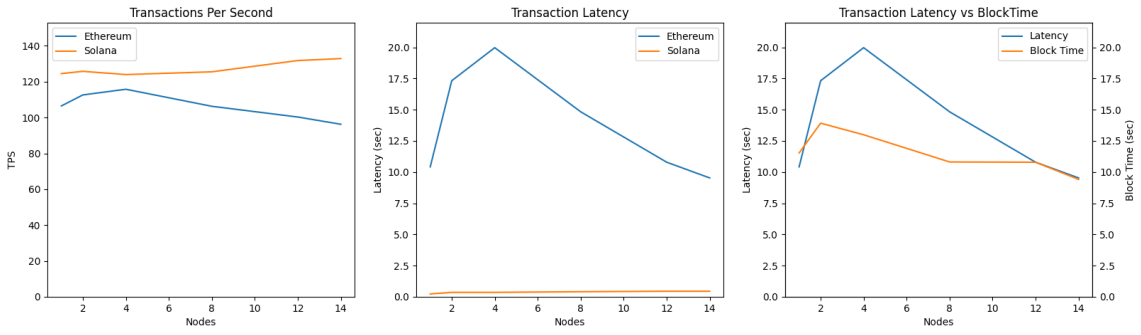
5.3.2 Επεκτασιμότητα

Στην παρούσα ενότητα θα εξετάσουμε την απόδοση των συστημάτων όσον αφορά την επεκτασιμότητα. Με άλλα λόγια, ξεκινάμε να δοκιμάζουμε το καθένα blockchain ξεχωριστά και καταγράφουμε την απόδοσή του καθώς μεταβάλλουμε το πλήθος των κόμβων που συμμετέχουν στο δίκτυο. Για αυτή την κατηγορία πειραμάτων τροφοδοτούμε το κάθε δίκτυο με 10000 συναλλαγές με ρυθμό 160 συναλλαγές ανά δευτερόλεπτο.

Τα πλήθη κόμβων που θα χρησιμοποιήσουμε είναι $1, 2, 4, 8, 12, 14$. Παρακάτω παρουσιάζουμε τα αποτελέσματα για κάθε benchmark:

DoNothing

Όπως έχουμε αναφέρει, το πείραμα DoNothing αφορά την ταχύτητα με την οποία επιτυγχάνεται το consensus στο δίκτυο. Από τα αποτελέσματα θα μπορέσουμε να συμπεράνουμε το συσχετισμό της ταχύτητας απόκρισης με τις συνθήκες του δικτύου:



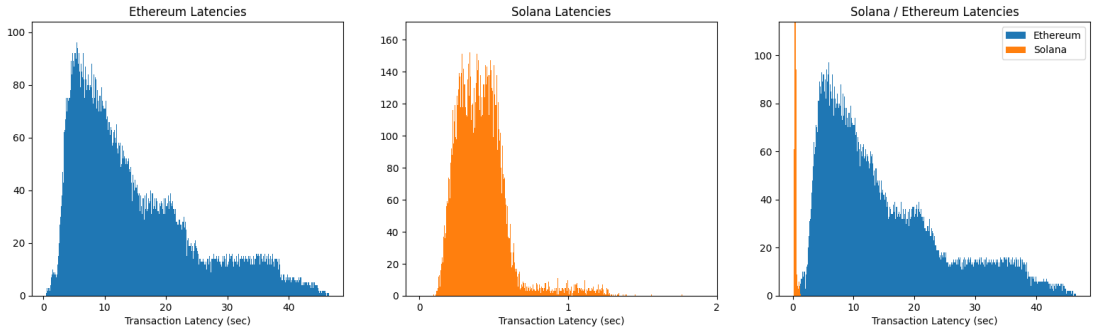
Σχήμα 5.1: *Αποτελέσματα για DoNothing*

Από τα παραπάνω αποτελέσματα βλέπουμε ότι το Solana διατηρεί ένα σχετικά σταθερό throughput το οποίο κυμαίνεται από 120 έως 130 συναλλαγές το δευτερόλεπτο, έχοντας μια ανοδική τάση. Αντιθέτως, το Ethereum φαίνεται να κυμαίνεται γύρω από τις 100 συναλλαγές ανά δευτερόλεπτο με μια πτωτική τάση. Επιπλέον, το Solana διατηρεί ένα σχετικά σταθερό χρόνο απόκρισης στις συναλλαγές, ο οποίος κυμαίνεται στα 0.5 δευτερόλεπτα.

Είναι ενδιαφέρον ότι για το Ethereum φαίνεται να υπάρχει μια μείωση στον χρόνο απόκρισης καθώς αυξάνονται οι κόμβοι, ωστόσο θα πρέπει να λάβουμε υπόψιν μας και τη

μέση τιμή του χρόνου που γίνεται mine κάθε block και παρατηρώντας το τρίτο γράφημα θα δούμε ότι το latency μειώθηκε καθώς μειώθηκε το block time. Αυτό είναι λογικό διότι όσο πιο γρήγορα γίνεται κάποιο block mined, τόσο πιο γρήγορα θα εξυπηρετηθούν και οι συναλλαγές στο εσωτερικό του. Επομένως, η επιτάχυνση στο consensus προήλθε από μια συμπτωματική πτώση στο block time. Τέτοιες διακυμάνσεις είναι ορατές ακόμαι και στο κύριο δίκτυο του Ethereum με κάποιο blocks να γίνονται mined ακόμα και σε 2 δευτερόλεπτα.

Ιδιαίρο ενδιαφέρον θα παρουσιάσει το επόμενο γράφημα, το οποίο μας δείχνει την κατανομή των χρόνων απόκρισης για τα δύο blockchains σε σύστημα 8 κόμβων:



Σχήμα 5.2: *Latency για το DoNothing*

Μπορούμε να παρατηρήσουμε ότι η πλειοψηφία των αποκρίσεων για το Solana βρίσκεται γύρω από τα 0.4 δευτερόλεπτα, ενώ για το Ethereum γύρω από τα 10. Ειδικά στο τρίτο γράφημα όπου γίνεται και η σύγκριση των δύο, παρατηρούμε ότι το Solana υπερβαίνει το Ethereum κατά πολύ.

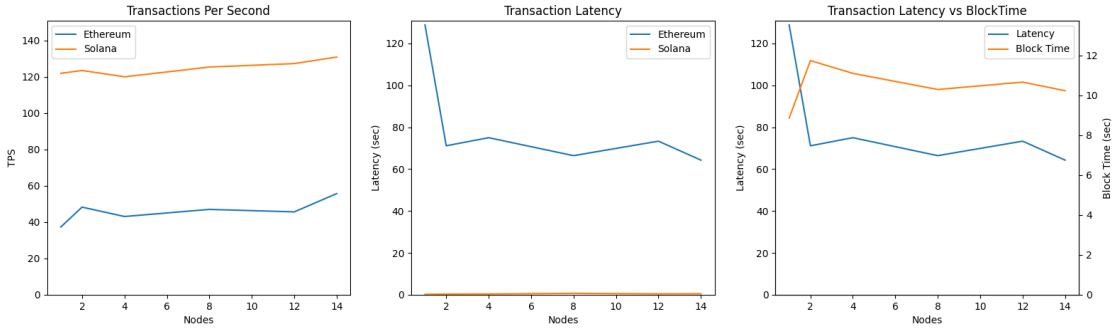
Από τα παραπάνω καταλήγουμε στο συμπέρασμα ότι το Solana μπορεί και επιτυγχάνει το consensus για μια συναλλαγή μέσα στον ονομαστικό του χρόνο. Ο χρόνος αυτός είναι ίσος με τον χρόνο του κάθε slot. Αντίστοιχα, το Ethereum έχει την πλειοψηφία των αποκρίσεων γύρω από τα 10 δευτερόλεπτα, ο οποίος είναι ο μέσος χρόνος block για το συγκεκριμένο πείραμα.

Η συμπεριφορά του Ethereum είναι φυσιολογική καθώς σύμφωνα με όσα περιγράψαμε στην ενότητά του, για να γίνει μια συναλλαγή αποδεκτή από το δίκτυο θα πρέπει πρώτα να περιέχεται στο block που έγινε mined. Αντιστοίχως και για το Solana, η συναλλαγή είναι επιτυχής όταν βρίσκεται στο slot που επαληθεύτηκε. Επομένως, είναι ασφαλές να καταλήξουμε ότι το consensus εξαρτάται από το block και το slot time για το Ethereum και το Solana αντιστοίχως.

KVStore

Το συμβόλαιο KVStore είναι ένα συμβόλαιο που διαβάζει και γράφει συνεχώς από τη μνήμη του συμβολαίου. Με τον τρόπο αυτό εξετάζουμε την ταχύτητα του κάθε blockchain στην ανάγνωση και εγγραφή. Παρακάτω παρατίθενται τα αποτελέσματα των πειραμάτων:

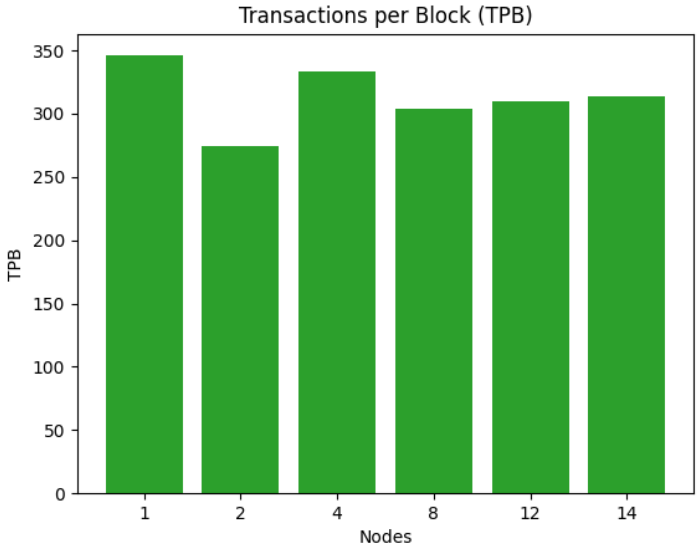
Παραπάνω βλέπουμε πως η συμπεριφορά του Solana παραμένει ίδια με το προηγούμενο πείραμα, δηλαδή μπορεί να εκτελεί αναγνώσεις και εγγραφές σε λογαριασμούς (όπως ανα-



Σχήμα 5.3: *Αποτελέσματα για το KVStore*

φέραμε για τον τρόπο υλοποίησης hashmaps μέσω PDAs) χωρίς να ζημειώνεται σε χρόνο. Αυτό μπορεί να ερμηνευτεί από την τεχνολογία του Cloudbreak η οποία προσφέρει γρήγορη πρόσβαση στους λογαριασμούς.

Από την άλλη πλευρά βλέπουμε ότι το τηρουγηπυτ του Ethereum παραμένει χαμηλό αλλά σταθερό. Η συμπεριφορά αυτή φανερώνει ότι υπάρχει και κάποια άλλη εξάρτηση στην απόδοση, η οποία δεν είναι άμεσα εμφανής. Η εξάρτηση αυτή είναι η ορισμένη τιμή του gasLimit. Όπως έχει αναφερθεί, το gasLimit του block υπαγορεύει ποιό είναι το μέγιστο gas που μπορεί να καταναλωθεί από το block. Επομένως, εφόσον το gasLimit είναι σταθερό, τότε το κάθε block μπορεί να περιέχει συγκεκριμένο πλήθος συναλλαγών. Παρακάτω φαίνεται το μέσω πλήθος συναλλαγών ανά block για το KVStore πείραμα:



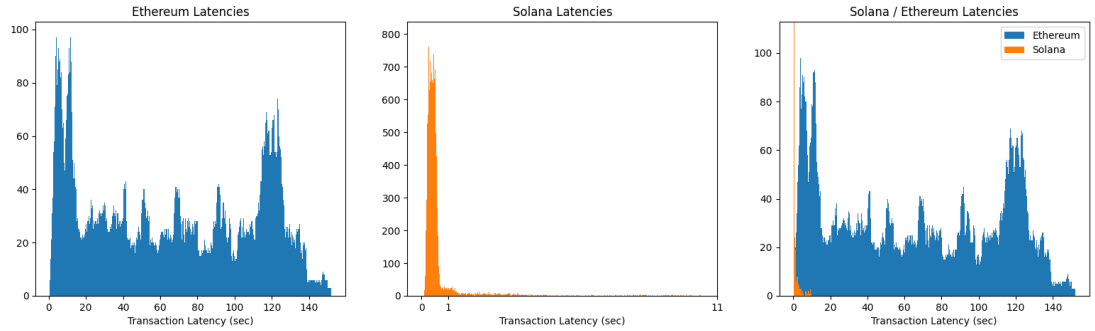
Σχήμα 5.4: *TPB για το KVStore*

Παρόλες τις αποκλίσεις, βλέπουμε ότι κάθε block έχει κατά μέση τιμή περίπου 300 συναλλαγές. Οι αποκλίσεις οφείλονται στο είδος των εμπεριεχόμενων συναλλαγών. Η λειτουργία ανάγνωσης δεν καταναλώνει το ίδιο gas με τη λειτουργία εγγραφής, επομένως λόγω της τυχαιότητας των συναλλαγών, το περιεχόμενο του block μπορεί να διαφέρει.

Επιπλέον, αξίζει να παρατηρήσουμε ότι το latency στο Ethereum έχει εκτοξευτεί κατά

πολύ περισσότερο από το block time, σε αντίθεση με πριν. Αυτό συμβαίνει καθώς το δίκτυο συσσωρεύει συναλλαγές αφού αυτές δεν μπορούν να εξυπηρετηθούν λόγω του περιορισμού του μεγέθους του block. Η συμπεριφορά του συγκεκριμένου πειράματος με μεταβλητό gasLimit εξετάζεται σε επόμενη ενότητα. Τέλος, αξίζει να αναφερθούμε στο spike του latency στους 2 κόμβους, το οποίο δικαιολογείται από το αντίστοιχο spike στο block time όπως φαίνεται από το τρίτο γράφημα.

Παρακάτω συγκρίνουμε και πάλι τα latencies μεταξύ των δύο blockchains για το πείραμα αυτό:

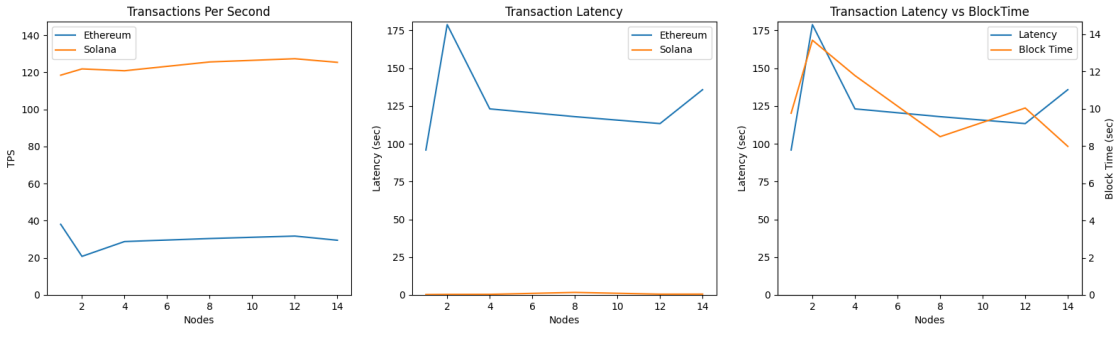


Σχήμα 5.5: *Latency για το KVStore*

Παρατηρούμε ότι το Solana έχει πλέον μεγαλύτερο εύρος στους χρόνους απόκρισής του, το οποίο θα μπορούσε να δικαιολογηθεί από την βαρύτητα των συναλλαγών αυτών. Ωστόσο, βλέπουμε ότι συγκριτικά με το Ethereum η διαφορά είναι εμφανής.

SmallBank

Το πείραμα SmallBank είναι συνδυαστικό πείραμα το οποίο μπορεί να περιέχει και πράξεις και αναγνώσεις και εγγραφές. Παρακάτω φαίνεται η συμπεριφορά του:

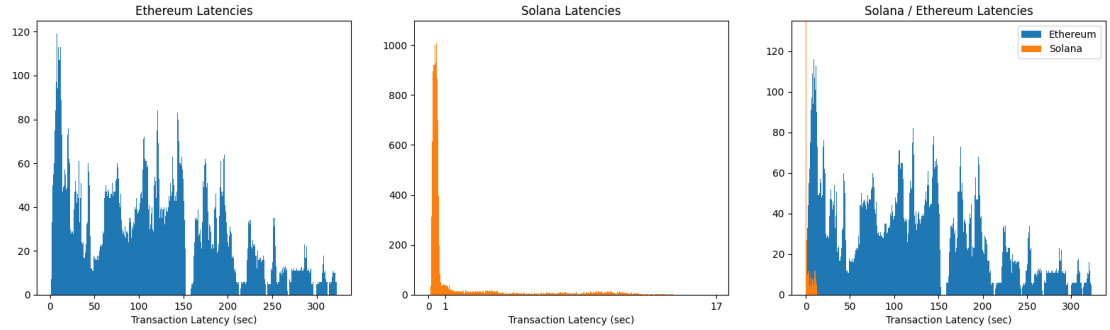


Σχήμα 5.6: *Αποτελέσματα για το SmallBank*

Εδώ βλέπουμε και πάλι το Solana να διατηρεί μια καλή, γρήγορη συμπεριφορά, ενώ το Ethereum έχει επίσης την ίδια ακριβώς συμπεριφορά με προηγουμένως. Η μικρή αύξηση στο throughput οφείλεται στα μικρότερα block times σε σχέση με πριν, καθώς τώρα τα blocks γίνονται mined πιο γρήγορα άρα έχουμε καλύτερη απόδοση. Το πείραμα σκοπίμως αφέθηκε με χαμηλό block time με σκοπό να αναδειχθεί η εξάρτηση του throughput από

αυτό. Επίσης, αξίζει να αναφέρουμε πως στο τρίτο γράφημα το latency και το block time ακολουθούν την ίδια συμπεριφορά και πλέον η εξάρτησή τους είναι προφανής.

Αντίστοιχα με τα προηγούμενα πειράματα, παραθέτουμε και το γράφημα με τα latencies για τους 8 κόμβους:



Σχήμα 5.7: *Latency για το SmallBank*

Αυτή τη φορά η βαρύτητα των πράξεων φαίνεται να επηρέασε τα latencies στο Solana. Επιπλέον, το χαμηλότερο block time μείωσε το εύρος των latencies στο Ethereum, γεγονός που εξηγήσαμε προηγουμένως. Παρόλα αυτά, είναι και πάλι εμφανές ότι το Solana παρουσιάζει πολύ καλύτερες επιδόσεις.

Συμπεράσματα

Συνολικά μπορούμε να παρατηρήσουμε ότι η επεκτασιμότητα του Ethereum εξαρτάται από διάφορους παράγοντες. Αρχικά, η πορεία του για απλές συναλλαγές με χαμηλό κόστος φαίνεται να είναι πτωτική, όπως είδαμε στο DoNothing πείραμα, γεγονός που μαρτυρά πως το δίκτυο έχει κόστη στην επικοινωνία και πιο συγκεκριμένα στις διαδικασίες του consensus. Όσο λιγότεροι είναι οι κόμβοι του τόσο πιοεύκολο είναι να έρθουν σε συμφωνία.

Από την άλλη, βλέπουμε επίσης πως αν οι συναλλαγές που έχει να εκτελέσει έχουν μεγάλο κόστος, τότε η απόδοσή του αρχίζει να εξαρτάται και από το gasLimit , ή πιο περιγραφικά, από το πόσες συναλλαγές μπορούν να χωρέσουν στο βλοσκ. Επομένως, μπορούμε να υποθέσουμε ότι με κατάλληλες ρυθμίσεις θα μπορούσε να επιτευχθεί μια καλύτερη απόδοση, ωστόσο στη συνέχεια θα αναφερθούμε σε αυτό το θέμα και θα εξετάσουμε αν υπάρχει κάποια χρυσή τομή.

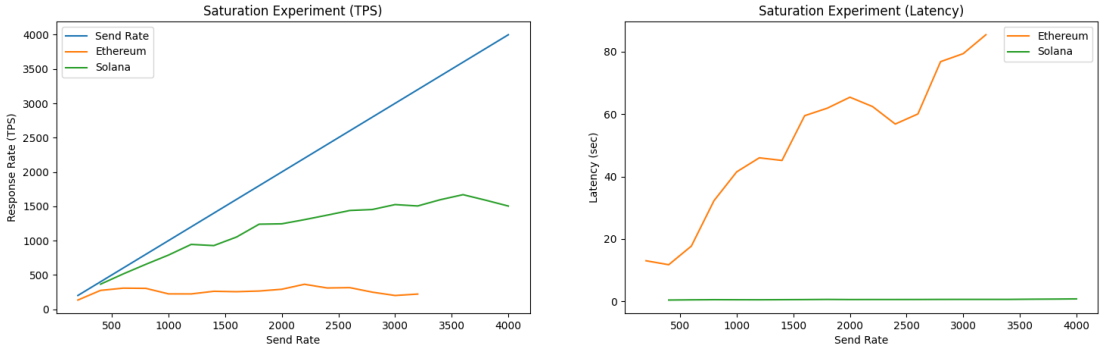
Όσον αφορά το Solana, είναι εμφανες ότι διατηρεί μια σταθερότητα στην απόδοσή του ανεξαρτήτως του πλήθους των κόμβων του δικτύου. Αυτό οφείλεται στο ότι οι συναλλαγές προωθούνται και τους επόμενους leaders έτσι ώστε αν ο τρέχν δεν καταφέρει να τις εκτελέσει, τότε αυτές να είναι έτοιμες προς εκτέλεση τον επόμενο. Αυτό είναι το Gulf Stream, όπως αναφερθήκαμε και στο προηγούμενο κεφάλαιο.

Φυσικά, βλέπουμε ότι η απόδοσή του επηρεάζεται από το είδος των συναλλαγών που επρόκειτο να εκτελέσει και ότι όσο πιο κοστοβόρες είναι τόσο πιο πολύ μειώνεται η ταχύτητα απόκρισης. Ωστόσο συνολικά είναι σαν να προσθέτουμε μια σταθερά στο latency του αρχικού πειράματος, το οποίο είναι λογικό καθώς οι συναλλαγές έχουν ήδη μπει στο pipeline του συστήματος και μετά από την καθυστέρηση των πρώτων, οι υπόλοιπες εκτελούνται με τις

αναμενόμενες αποκρίσεις.

5.3.3 Κορεσμός

Στη συνεχεια, εκτελέσαμε ένα πείραμα στο οποίο προσπαθούμε να φτάσουμε το κάθε σύστημα στα όριά του, στέλνοντας ένα σταθερό αριθμό από 32000 συναλλαγές ενώ ταυτόχρονα αυξάνουμε το ρυθμό αποστολής τους. Οι συναλλαγές είναι απλές μεταφορές του κάθε νομίσματος μεταξύ λογαριασμών του συστήματος, επομένως περιμένουμε ότι θα δαπανούν τους λιγότερο δυνατούς πόρους. Τα αποτελέσματα φαίνονται παρακάτω:



Σχήμα 5.8: *Κορεσμός Συστημάτων*

Στο πρώτο γράφημα η μπλε γραμμή αναπαριστά το θεωρητικό τηρουγηπυτ που θα είχε το σύστημα αν ήταν ιδανικό και είχε χρόνο απόκρισης 0 δευτερόλεπτα. Βλέπουμε πως το Solana ακολουθεί αρκετά καλά μέχρι περίπου τις 1700 συναλλαγές το δευτερόλεπτο, ενώ μετα σταθεροποιείται στις 1500 συναλλαγές ανά δευτερόλεπτο. Αντιθέτως, βλέπουμε το Ethereum να μην μπορεί να ακολουθήσει ούτε ακόμα και στους πιο αργούς ρυθμούς αποστολής. Αυτόείναι αποτέλεσμα του ότι οι συναλλαγές δεν εκτελούνται απευθείας αλλά συγκενρώνονται με σκοπό να ενσωματωθούν στο επόμενο block και άρα επέρχεται αυτή η καθυστέρηση.

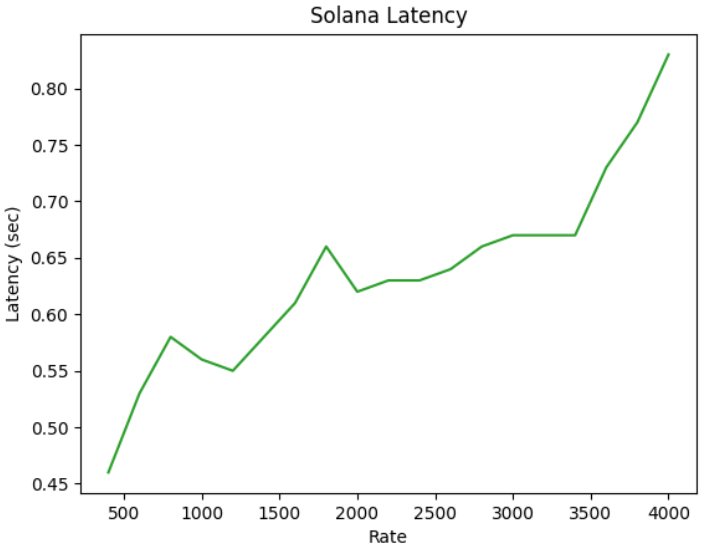
Όσον αφορά το latency βλέπουμε πως στο Ethereum συνεχώς αυξανεται, γεγονός που επιβεβαιώνει και πάλι τον ισχυρισμό μας ότι οι συναλλαγές συκεντρώνονται αντί να εκτελούνται. Αν αναλογιστεί κανείς ότι ακόμα και με χαμηλό gas, δεν μπορούν να χωρέσουν απεριόριστες συναλλαγές μέσα στο βλοσκ, τότε είναι πολύ λογική αυτή η μεγάλη αύξηση στο latency.

Φυσικά, και το Solana παρουσιάζει αύξηση στο latency του, ωστόσο αυτή δεν φαίνεται λόγω της πολύ μικρής της απόκλισης. Παρακάτω παρουσιάζεται το γράφημα του latency για το Solana μεγενθυμένο:

Εν τέλει, η αύξηση του Solana διατηρείται ακόμα κάτω από το 1 δευτερόλεπτο.

Συμπεράσματα

Θεωρούμε ότι η κακή απόδοση στο Ethereum οφείλεται στην αρχιτεκτονική του και όλα τα εμπόδια που προαναφέραμε, όμως για το Solana θεωρούμε ότι περιορίζεται λόγω προδιαγραφώ του συστήματος. Αυτό δικαιολογείται από την πολύ μεγάλη απόκλιση που έχει

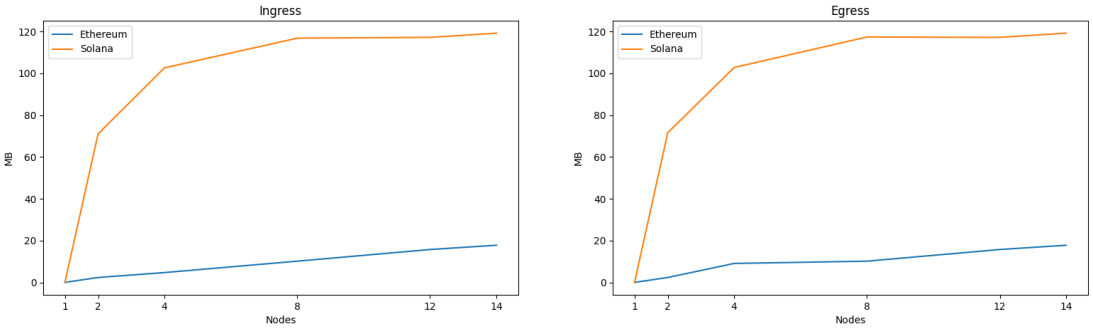


Σχήμα 5.9: *Latency για Solana*

το σύστημά μας από τις προτεινόμενες προδιαγραφές και από το γεγονός ότι το Solana δεν μπορούσε να ανταποκριθεί στο ρυθμό αποστολής των συναλλαγών και έπαυε να λειτουργεί. Επιπλέον, αν λάβουμε υπόψιν μας ότι η ομάδα του Solana δηλώνει ότι μπορεί να διεκπεραιωθεί ρυθμός έως και 70000 συναλλαγές το δευτερόλεπτο, τότε γίνεται πιο ξεκάθαρο το ζήτημα των υποδομών μας.

5.3.4 Input/Output

Μια άλλη μέτρηση που καταγράψαμε είναι η ποσότητα των δεδομένων που εισήλθαν και εξήλθαν κατά μέσο όρο από όλους τους κόμβους του κάθε συστήματος. Ορίζουμε σαν egress τα δεδομένα που στάλθηκαν και ως ingress τα δεδομένα που λήφθηκαν από τον κάθε κόμβο. Παρακάτω έχουμε τα αποτελέσματα και τις παρατηρήσεις μας:



Σχήμα 5.10: *Input/Output για KVStore*

Παρατηρούμε ότι το Solana διαχειρίζεται πολύ μεγαλύτερο πλήθος δεδομένων όσον αφορά το δίκτυο, γεγονός που επίσης δικαιολογεί και τις τεράστιες απαιτήσεις δικτύου που έχει. Για να δώσουμε μια ικανοποιητική εξήγηση για την παραπάνω μέτρηση θα πρέπει να προσπαθήσουμε να σκεφτούμε τί ενέργειες εκτελεί το κάθε blockchain σε σχέση με το

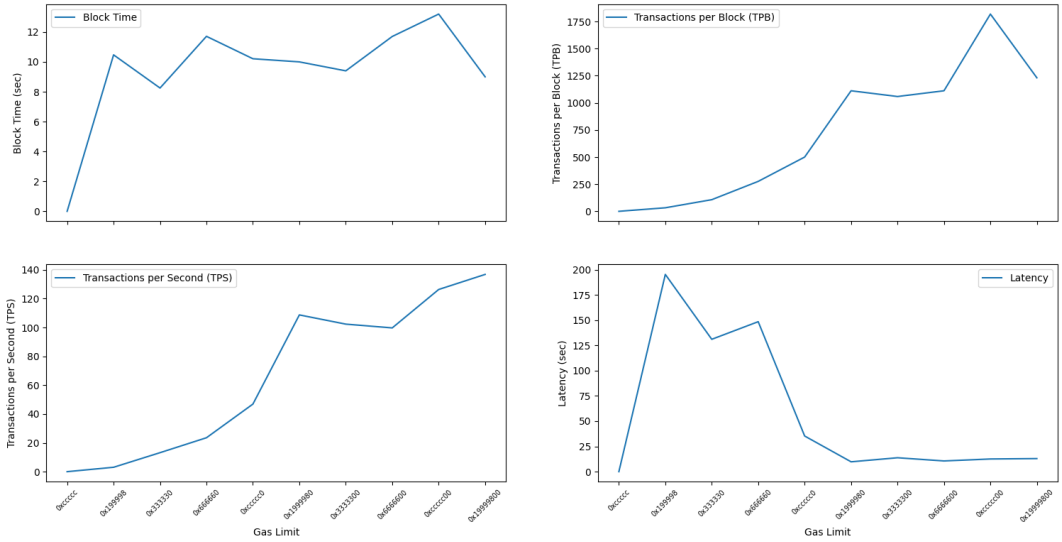
δίκτυο.

Αρχικά, ένας κόμβος στο Ethereum λαμβάνει συναλλαγές από τους χρήστες αλλά και από τους υπόλοιπους κόμβους. Οι συναλλαγές αυτές στη συνέχει επεξεργάζονται και εφόσον ο κόμβος καταφέρει να κανει mine το block, τότε το δημοσιεύει στο δίκτυο και γίενται το ϛονσενσυς. Επιπλέον, κάθε συναλλαγή που λαμβάνει την προωθεί στο δίκτυο. Όλα τα παραπάνω γίνονται με τον ίδιο τρόπο και στο Solana, επομένως αν υποθέσουμε ότι το μέγεθος των συναλλαγών έιναι το ίδιο, μπορούμε να πούμε ότι σε αυτό το σημείο και τα δύο δίκτυα βρίσκονται στην ίδια θέση.

Ωστόσο, το Ethereum διεξάγει ψηφοφορία για το κάθε block κάθε φόρα που αυτό γίνεται mine και επομένως αν δούμε τα αποτελέσματα για το KVStore, τότε μπορούμε να κάνουμε την παραδοχή ότι το block time ισούται περίπου με 10 δευτερόλεπτα. Όμως το Solana διεξάγει το validation κάθε 400ms, όπου αυτό είναι και το slot time, άρα έχουμε περίπου 25 φορές περισσότερες ψηφοφορίες. Προφανώς τα πακέτα για κάθε δίκτυο δεν θα είναι ίσα, επομένως μπορούμε να πούμε ότι η διαφορά των 10-20 φορών δικαιολογείται από αυτή τη διαφορά στο πλήθος ψηφοφοριών.

5.3.5 Ethereum - Μεταβλητό Gas

Σε αυτό το πείραμα εκτελέσαμε με 8 κόμβους το KVStore συμβόλαιο με σταθερό ρυθμό 160 συναλλαγές το δευτερόλεπτο, όπως στην πρώτη υποενότητα. Η διαφορά τώρα είναι, ότι για κάθε τρέξιμο του πειράματος αυξάνουμε το gas limit του block και παρατηρούμε πώς συμπεριφέρεται το δίκτυο. Παρακάτω παρουσιάζουμε τις μετρήσεις μας και εξηγούμε τα αποτελέσματα:



Σχήμα 5.11: *Μεταβλητό gas για KVStore*

Block Time

Παρατηρούμε ότι παρόλο που το γας αυξάνεται, το block time παραμένει σετικά σταθερό και κυμαίνεται στις φυσιολογικές του τιμές. Αυτό είναιφυσιολογικό μιας και ο κάθε miner

θα πρέπει να ικανοποιήσει δύο κριτήρια κάθε φορά. Το ένα είναι να συγκεντρώσει τόσες συναλλαγές, έτσι ώστε το συνολικό τους gas να πλησιάζει το gas limit του block για να μεγιστοποιήσει τα κέρδη του από τα φεες. Ωστόσο, πρέπει να ικανοποιήσει και το target block time, επομένως θα προσπαθεί να κάνει mine το κάθε block σε χρόνο κοντά στο χρόνο στόχο. Έτσι, αυτό παραμένει ε ένα λογικό εύρος τιμών.

Transactions Per Block

Όπως έχει προκύψει από την προηγούμενη ανάλυση είναι φυσιολογικό ότι όταν ο κόμβος μπορεί να συγκεντρώσει περισσότερες συναλλαγές (εφόσον αυτές είναι διαθέσιμες), τότε θα το κάνει ώστε να μεγιστοποιήσει τα κέρδη του. Επομένως, βλέπουμε το κάθε block να έχει όλο και περισσότερες συναλλαγές κάθε φορά που το gas limit του το επιτρέπει.

Transactions Per Second

Το throughput αυξάνεται όσο οι συναλλαγές μέσα στο block αυξάνονται. Όταν ένα block έχει περισσότερες συναλλαγές, τότε το σύνολο των συναλλαγών θα εκτελεστεί σε λιγότερα block και εφόσον το block time ορίζει το συνολικό χρόνο εκτέλεσης, τότε το throughput θα αυξάνεται.

Latency

Το latency ολοένα και μειώνεται, ενώ το βλέπουμε στα μεγάλα gas limits να παραμένει σταθερό. Αυτό συμβαίνει διότι μειώνεται εώς ώτου αγγίξει το block time, καθώς περεταίρω βελτίωση δεν μπορεί να υπάρξει. Αυτό συμβαίνει γιατί για να επικυρωθεί μια συναλλαγή θα πρέπει πρώτα να έχει συμπεριληφθεί σε κάποιο mined block.

Συμπέρασμα

Θα μπορούσε κανεί να ισχυριστεί ότι αφού μπορούμε να μεγαλώσουμε το gas limit του block και να επιτύχουμε μεγαλύτερο throughput, τότε θα έπρεπε να το ορίσουμε στη μέγιστη τιμή. Όμως αυτό είναι λάθος, καθώς μεγάλο gas limit σημαίνει ότι κάθε block θα περιέχει περισσότερες συναλλαγές με αποτέλεσμα να χρειάζεται περισσότερο χρόνο ώστε να γίνει επικύρωσή τους. Αυτό με τη σειρά του έχει ως συνέπεια το block να γίνει καθυστερημένα mined με αποτέλεσμα να μην προλάβει να μπει στην αλυσίδα και να γινει uncle block αυξάνοντας την πουπλοκότητα του δικτύου.

Επιπλέον, περισσότερες συναλλαγές σημαίνουν και περισσότεροι πόροι για την επικύρωσδή τους, άρα μια τέτοια κίνηση θα οδηγούσε τους κόμβους με μεγαλύτερη υπολογιστική ισχύ να κάνουν mine ολοένα και περισσότερα blocks με αποτέλεσμα να καταλύετα η έννοια του αποκεντρωμένου δικτύου και η δύναμη να συγκεντρώνεται στα χέρια των ισχυρότερων κόμβων.

Κατα συνέπεια, ο ορισμός της τιμή του gas limit θα πρεπει να γίνεται με προσοχή και να προσαρμόζεται συνεχώς ώστε να αποφεύγονται τέτοιες συνθήκες. Εξάλλου, το Ethereum προσαρμόζει το gas limit του block αναλόγως τη μέχρι εκείνη στιγμή κατάστασή του ώστε να μεγιστοποιεί σταδιακά το throughput του συστήματος.

5.3.6 CPUHeavy & IOHeavy

Τα συμβόλαια της κατηγορίας αυτής αποτελούν συμβόλαια με πολλές υπολογιστικές πράξεις. Πιο συγκεκριμένα, το συμβόλαιο CPUHeavy αφορά την εκτέλεση ενός αλγορίθμου ταξινόμησης πινάκων και πιο συγκεκριμένα υλοποιεί τον αλγόριθμο quicksort. Από την άλλη μεριά, το συμβόλαιο IOHeavy εκτελεί έναν αλγόριθμο όπου καταχωρεί πάρα πολλά δεδομένα στη μνήμη του smart contract μέσω μιας συνάρτησης και στη συνέχεια μέσω μιας άλλης συνάρτησης εκτελεί αναγνώσεις επί του πίνακα αυτού.

Παρατηρήσεις

Το συγκριτικό αυτό πείραμα δεν ήταν δυνατόν να εκτελεστεί και οι λόγοι αφορούν κυρίως το Solana. Στον κώδικα του Solana υπάρχει μια παράμετρος όπου ονομάζεται MAX_COMPUTE_UNIT_LIMIT η οποία παίζει παρόμοιο ρόλο με το gas limit στο Ethereum. Το ζήτημα με την παράμετρο αυτή είναι ότι η τιμή της είναι ορισμένη σε πολύ χαμηλά επίπεδα με αποτελέσμα τα συμβόλαια να μην μπορούν να εκτελεστούν για μεγάλες εισόδους. Επομένως, η σύγκριση σε πειραματικό επίπεδο θα ήταν ανέφικτη καθώς στο Ethereum ο χρήστης μπορεί να ορίσει μόνος του το πλήθος των υπολογιστικών πόρων που επιθυμεί να καταναλώσει και άρα μπορεί να τους θέσει σε πολλά υψηλά επίπεδα, ενώ στο Solana κάτι τέτοιο δεν μπορεί να συμβεί.

Επομένως, από τα παραπάνω συμπεραίνουμε ότι το Solana είναι ένα blockchain που δεν αποσκοπεί στην εκτέλεση ¨βαριών¨ υπολογιστικών πράξεων. Κάτι τέτοιο μπορεί να ερμηνεύσει και τους γρήγορους χρόνους που αποδίδει καθώς δεν δέχετε συναλλαγές που μπορεί να επιβαρύνουν το latency του. Αυτή η παράμετρος είναι πολύ σημαντική καθώς θέτει σαφείς περιορισμούς στις δυνατότητες που μπορεί κανείς να δώσει στην εφαρμογή του.

Φυσικά κάποιος θα μπορούσε να εκτελέσει πολύπλοκους υπολογισμούς στο Solana χωρίζοντας κατάλληλα τη δομή του αλγορίθμου που επιθυμεί να ακολουθήσει και στέλνοντας επιμέρους πιό απλές συναλλαγές που θα μπορούσαν να εκτελεστούν αυτόνομα. Κάτι τέτοιο ωστόσο δεν μπορεί να μας δώσει μια αντικειμενική εικόνα του δικτύου και για το λόγο αυτό, αυτά τα πειράματα δεν εκτελέστηκαν.

5.4 Συμπεράσματα

Με μια πρώτη ματιά θα μπορούσε κανείς να παρατηρήσει μια χαωτική διαφορά στις επιδόσεις των δύο εξεταζόμενων blockchain συστημάτων. Πράγματι, το Solana φαίνεται να προσφέρει πολύ πιο εντυπωσιακούς χρόνους απόκρισης στις συναλλαγές αλλά ακόμα να προσφέρει throughput κοντά στον ρυθμό αποστολής των δεδομένων. Δεν είναι τυχαίο άλλωστε που οι προγραμματιστές του το χαρακτηρίζουν ως μια streaming πλατφόρμα.

Παρόλα αυτά, όταν κανείς κληθεί να πάρει μια σχεδιαστική απόφαση σχετικά με την πλατφόρμα πάνωστην οποία θα θελήσει να δομήσει την εφαρμογή του, θα πρέπει να λάβει υπόψιν του τον πολύ σημαντικό περιορισμό των υπολογιστικών πόρων που θέτει το Solana. Ο παραπάνω περιορισμός είναι σημαντικός καθώς μια εφαρμογή πρέπει να είναι ανταποκρίσιμη σε κάθε είδους συνθήκες και να μην διακόπτει τη λειτουργία της για απρόβλεπτους

λόγους, όπως για παράδειγμα την εισαωγωγή μιας τιμής η οποία μπορεί να οδηγήσει το δίκτυο στην κατανάλωση όλων των πόρων της συναλλαγής.

Συμπεραίνουμε λοιπόν ότι η απόφαση για το ποιό blockchain θα πρέπει να χρησιμοποιήσουμε για την εαρμογή μας δεν είναι τόσο απλή και θα πρέπει να λάβουμε υπόψιν μας το τί αλλά και το πώς θα εκτελεστούν οι πράξεις του συμβολαίου μας. Φυσικά, αν πρόκειτα να εφαρμόζουμε απλή λογική στα συμβόλαιά μας χωρίς corner cases, τότε το Solana φαίνεται μια πιό λογική επιλογή καθώς θα δώσει στην εφαρμογή μας την μέγιστη ταχύτητα.

Μέρος III

Επίλογος

Κεφάλαιο 6

Επίλογος

Ελπίζουμε η παρούσα εργασία να ήταν αποσαφηνιστική όσον αφορά την περιγραφή της τεχνολογίας του blockchain αλλά και της εφαρμογής της σε δύο από τα πιο γνωστά συστήματα, το Ethereum και το Solana. Ο σημερινός ρυθμός των γεγονότων στον τομέα της Πληροφορικής απαιτεί συνεχή επαγρύπνιση και παρακολούθηση των εξελίξεων ώστε να μπορεί κανείς να ανταποκριθεί στις ανάγκες των καιρών. Η τεχνολογία του blockchain ενδέχεται να είναι καταλυτική για την πορεία των επικοινωνιών αλλά και τη δομή κοινωνικών και οικονομικών οργανισμών στο μέλλον και η γνώση της μόνο καλό μπορεί να αποφέρει στον κάτοχό της.

Όσον αφορά την παρούσα εργασία, θα μπορούσε κανείς να αναρωτηθεί ποιό είναι το τελικό συμπέρασμα, ποιό blockchain σύστημα είναι ¨καλύτερο¨ και μπορεί να ανταπεξέλθει στις απαιτήσεις των χρηστών. Καλώ ή κακώς, απόλυτη απάντηση δεν μπορεί να δοθεί, όπως δεν μπορεί να δοθεί απόλυτη απάντηση σε πολλά από τα προβλήματα της Πληροφορικής.

Η παρούσα εργασία παρουσιάζει όλες τις όψεις των δύο συστημάτων και κάνει μια συγκριτική ανάλυσή τους. Το Ethereum έχει υπάρξει ένας από τους μεγαλύτερους πυλώνες του κλάδου των blockchain, μέσω αυτού γίνονται καθημερινά αμέτρητες συναλλαγές ή ακόμα και πιο γενικές οικονομικές δραστηριότητες. Εισήγαγε νέες έννοιες, όπως τα smart contracts τα οποία άλλαξαν μια για πάντα τον χαρακτήρα του blockchain. Ωστόσο, η μεγάλη δημοφιλία προσελκύει πολλούς χρήστες και η αντιμετώπιση του προβλήματος της εξυπηρέτησης δεν είναι πάντα εύκολη.

Από την άλλη πλευρά, το Solana δείχνει ένα πολλά υποσχόμενο νέο σύστημα με επιδόσεις εφάμιλλες με αυτές μεγάλων τεχνολογικών επιχειρήσεων. Η τεχνολογία του φαίνεται δελεαστική και ταυτόχρονα πρωτόγνωρη, παρουσιάζει δυσκολίες στην εκμάθηση και απαιτεί μια πιο προσεκτική προσέγγιση σε σχέση με το Ethereum.

6.1 Το πρόβλημα της επιλογής

Όλα καταλήγουν στο εξής ερώτημα, ποιά τεχνολογία να επιλέξω από τις δύο για την εφαρμογή μου· Η αλήθεια είναι ότι υπάρχουν αρκετοί παράγοντες που πρέπει να λάβει κανείς υπόψιν του για να πάρει μια απόφαση.

Αρχικά, ο χρήστης που θα ήθελε μια εφαρμογή με υψηλές αποδόσεις και ταυτόχρονα χαμηλά κόστη θα μπορούσε να στραφεί στο Solana λόγω των χαμηλών fees που έχει. Ωστόσο, αν η εφαρμογή του θα πρέπει να κάνει μια πιο απαιτητική διαχείριση στην μνήμη και να

είναι πιο εύκολα συντηρήσιμη, τότε ίσως θα πρέπει να στραφεί προς το Ethereum.

Επιπλέον, ένας προγραμματιστής θα πρέπει να λαμβάνει σοβαρά υπόψιν του το μέγεθος της κοινότητας στην οποία θα εμπλακεί. Η εργασία αυτή απέδειξε ότι η κοινότητα του blockchain είναι ακόμα μικρή και η υποστήριξη που μπορεί να βρει κανείς μπορεί να είναι δύσκολη. Πιο συγκεκριμένα, η κοινότητα του Solana είναι ακόμα άγουρη και πολλές φορές ακόμα και για ένα απλό βήμα μπορεί κανείς να ξοδέψει μέρες αναζήτησης μέχρι να βρει την απάντηση. Σε έναν όχι τόσο έντονο βαθμό θα μπορούσε να πει κανείς ότι το ίδιο ισχύει και για το Ethereum. Σε κάθε περίπτωση όμως, αν κάποιος θα ήθελε να χτίσει μια εφαρμογή με περισσότερη ασφάλεια, τότε θα επέλεγε το Ethereum.

Από την άλλη πλευρά, από τεχνικής απόψεως το Solana φαίνεται να υπερέχει, φαίνεται να έχει καλύτερες επιδόσεις και είναι πολλά υποσχόμενο. Όμως, η ιστορία του έχει δείξει ότι δεν είναι τόσο εύρωστο, όπως πρόσφατα που υπήρξε ένα πάγωμα λόγω του μεγάλου όγκου ταυτόχρονων συναλλαγών που μπήκαν στο δίκτυο.

6.1.1 Συμπέρασμα

Εν ολίγοις, η επιλογή ενός συστήματος για τη διεκπεραίωση μιας εργασίας δεν διαφέρει περισσότερο από την επιλογή μιας γλώσσας προγραμματισμού για την υλοποίηση ενός αλγορίθμου. Όλα καταλήγουν στην προτίμηση του εκτελεστή και το μόνο που θα μπορούσα να συμβουλέψω κάποιον που βρίσκεται στο δίλημμα της επιλογής θα ήταν να φροντίσει ώστε να αποκτήσει γνώση από όσα περισσότερα ερεθίσματα μπορεί να έχει.

6.1.2 Μελλοντική δουλειά

Μερικά ζητήματα που εμφανίστηκαν μπροστά μου κατά την εκπόνηση της εργασίας αυτής και μου κέντρισαν το ενδιαφέρον είναι τα παρακάτω. Θεωρώ ότι μόνο κερδισμένος μπορεί να βγεί κανείς από αυτή την αναζήτηση:

- Μελέτη σε αλγορίθμους για consensus χωρίς την ανάγκη του mining
- Μελέτη της δυνατότητας αποδοτικής αποθήκευσης των δεδομένων του blockchain διατηρώντας το Proof of Replication
- Μελέτη της αρχιτεκτονικής του TowerBFT και της αποδοτικότητάς του

Βιβλιογραφία

[1] Leslie Lamport, Robert Shostak και Marshall Pease. *The Byzantine generals problem*. *Concurrency: the works of leslie lamport*, σελίδες 203–226. 2019.

[2] Satoshi Nakamoto. *Bitcoin: A peer-to-peer electronic cash system*. *Decentralized Business Review*, σελίδα 21260, 2008.

[3] W Ethereum. *Ethereum Whitepaper*. *Ethereum. URL: https://ethereum. org [accessed 2020-07-07]*, 2014.

[4] Micah Dameron. *Beigepaper: an ethereum technical specification*. *Ethereum Project Beige Paper*, 2018.

[5] Anatoly Yakovenko. *Solana: A new architecture for a high performance blockchain v0. 8.13*. *Whitepaper*, 2018.

[6] Juan Benet, David Dalrymple και Nicola Greco. *Proof of replication*. *Protocol Labs, July*, 27:20, 2017.

[7] *Solana Archivers*. https://medium.com/solana-labs/replicators-solanas-solution-to-petabytes-of-blockchain-data-storage-ef79db053fa1.

[8] *Solana TowerBFT*. https://medium.com/solana-labs/tower-bft-solanas-high-performance-implementation-of-pbft-464725911e79.

[9] *Solana Turbine*. https://medium.com/solana-labs/turbine-solanas-block-propagation-protocol-solves-the-scalability-trilemma-2ddba46a51db.

[10] *Solana Gulf Stream*. https://medium.com/solana-labs/gulf-stream-solanas-mempool-less-transaction-forwarding-protocol-d342e72186ad.

[11] *Solana SeaLevel*. https://medium.com/solana-labs/sealevel-parallel-processing-thousands-of-smart-contracts-d814b378192.

[12] *Solana Pipelining*. https://medium.com/solana-labs/pipelining-in-solana-the-transaction-processing-unit-2bb01dbd2d8f.

[13] *Solana CloudBreak*. https://medium.com/solana-labs/cloudbreak-solanas-horizontally-scaled-state-architecture-9a86679dcbb1.

[14] Tien Tuan Anh Dinh, Ji Wang, Gang Chen, Rui Liu, Beng Chin Ooi και Kian Lee Tan. *Blockbench: A framework for analyzing private blockchains*. *Proceedings of the 2017 ACM international conference on management of data*, σελίδες 1085–1100, 2017.