ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ
ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ανίχνευση Επιθέσεων από Εσωτερικές Απειλές με Χρήση Τεχνολογίας Blockchain

Κουτσολέλος Νικόλαος

ΑΜ: 713 45517

Επιβλέπουσα Καθηγήτρια:

Ιωάννα Καντζάβελου

Αθήνα, Οκτώβριος 2023

UNIVERSITY OF WEST ATTICA
SCHOOL OF ENGINEERING
DEPARTMENT OF INFORMATICS AND COMPUTER ENGINEERING

Thesis

Insider Attack Detection Using Blockchain

Koutsolelos Nikolaos

Registration Number: 713 45517

Supervisor:

Ioanna Kantzavelou

Athens, October 2023

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ανίχνευση Επιθέσεων Από Εσωτερικές Απειλές Με Χρήση Τεχνολογίας Blockchain

Κουτσολέλος Νικόλαος
Α.Μ. 713 45517

Εισηγητής:

Ιωάννα Καντζάβελου, Επίκουρη Καθηγήτρια

Ioanna Kantzavelou
Digitally signed by Ioanna Kantzavelou
Date: 2023.10.23 10:22:19 +03'00'

Εξεταστική Επιτροπή:

Vasileios Mamalis
Digitally signed by Vasileios Mamalis
Date: 2023.10.26 00:30:42 +03'00'

Βασίλειος Μάμαλης, Καθηγητής

Antonios Bogris
Digitally signed by Antonios Bogris
Date: 2023.10.24 17:38:45 +03'00'

Αντώνιος Μπόγρης, Καθηγητής

Ημερομηνία εξέτασης: 13/10/2023

ΔΗΛΩΣΗ ΣΥΓΓΡΑΦΕΑ ΔΙΠΛΩΜΑΤΙΚΗΣ ΕΡΓΑΣΙΑΣ

Ο κάτωθι υπογεγραμμένος Νικόλαος Κουτσολέλος του Γεωργίου, με αριθμό μητρώου 713 45517 φοιτητής του Πανεπιστημίου Δυτικής Αττικής της Σχολής Μηχανικών του Τμήματος Μηχανικών Πληροφορικής και Υπολογιστών, δηλώνω υπεύθυνα ότι:

«Είμαι συγγραφέας αυτής της διπλωματικής εργασίας και κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, οι όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων, είτε ακριβώς είτε παραφρασμένες, αναφέρονται στο σύνολό τους, με πλήρη αναφορά στους συγγραφείς, τον εκδοτικό οίκο ή το περιοδικό, συμπεριλαμβανομένων και των πηγών που ενδεχομένως χρησιμοποιήθηκαν από το διαδίκτυο. Επίσης, βεβαιώνω ότι αυτή η εργασία έχει συγγραφεί από μένα αποκλειστικά και αποτελεί προϊόν πνευματικής ιδιοκτησίας τόσο δικής μου, όσο και του Ιδρύματος.
Παράβαση της ανωτέρω ακαδημαϊκής μου ευθύνης αποτελεί ουσιώδη λόγο για την ανάκληση του διπλώματός μου».

Ο Δηλών,

Νικόλαος Κουτσολέλος

Ευχαριστίες

Η παρούσα διπλωματική εργασία ολοκληρώθηκε μετά από επίμονες προσπάθειες, σε ένα ενδιαφέρον γνωστικό αντικείμενο. Την προσπάθεια μου αυτή υποστήριξε η επιβλέπουσα καθηγήτρια μου κυρία Καντζάβελου, την οποία θα ήθελα να ευχαριστήσω.

Περίληψη

Σε ένα υπολογιστικό σύστημα το οποίο δεν είναι συνδεδεμένο με τον έξω κόσμο (διαδίκτυο), οι απειλές είναι πιο πιθανό να πραγματοποιηθούν από κάποιον που έχει δικαιώματα στο σύστημα αυτό, όπως για παράδειγμα κάποιον διαχειριστή ο οποίος μπορεί στη συνέχεια να διαγράψει, να τροποποιήσει ή να καταστρέψει τα ενοχοποιητικά στοιχεία από το σύστημα (logs) κάνοντας την αναγνώριση, την ιχνηλάτηση, και αργότερα την ανάλυση της επίθεσης αρκετά πιο δύσκολη. Για αυτό το λόγο θα δοκιμαστεί η χρησιμότητα της τεχνολογίας blockchain, δηλαδή κατανεμημένη εγγραφή των αλλαγών, την οποία δεν θα μπορεί να τροποποιήσει μόνος του ο επιτιθέμενος χωρίς αυτό να γίνει αντιληπτό, ως εναλλακτικός τρόπος καταγραφής. Θα δημιουργηθεί μια βάση δεδομένων, οι αλλαγές της οποίας θα καταγράφονται (χρήστης, ώρα, αλλαγή, κτλ) και με τη χρήση του συστήματος που θα αναπτυχθεί θα καταχωρούνται σε ένα blockchain προκειμένου είτε να ειδοποιηθούν οι αρμόδιοι, στην περίπτωση που κάποιος προσπαθήσει να αλλάξει τα στοιχεία που καταγράφονται, είτε να είναι πιο εύκολη η μελλοντική ανάλυση της επίθεσης (ποιός το έκανε, τι άλλαξε, ποτέ, κτλ).

Abstract

In a computer system that is not connected to the outside world (internet), threats are more likely to be carried out by someone who already has rights to the system, such as an administrator who can then delete, modify or destroy the incriminating data from the system for example the logs, making the identification, tracing, and later analysis of the attack much more difficult. For this reason, the usefulness of blockchain technology, i.e. distributed recording of changes which the attacker will not be able to modify himself without being noticed, will be tested as an alternative way of logging. A database will be created, the changes of which will be logged (user, time, change, etc.) using a blockchain in order to either notify the authorities, in the event that someone tries to tamper with the system, or to make the future analysis of the attack easier (who did it, what he changed, when, etc.).

Πίνακας Περιεχομένων

Πίνακας Σχημάτων - Εικόνων

Κεφάλαιο 1. Συστήματα Ανίχνευσης Εισβολών

1.1 Εισαγωγή

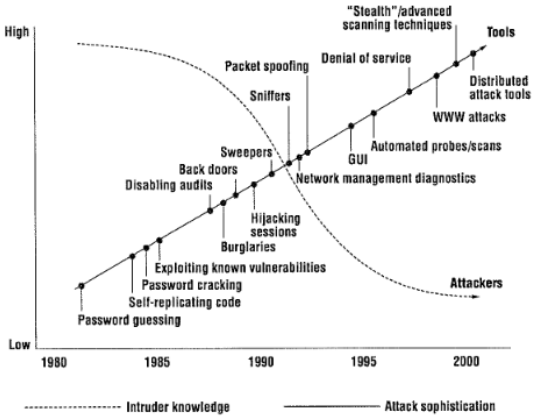
Σήμερα, όπου ολοένα και περισσότεροι οργανισμοί και επιχειρήσεις εξαρτώνται από την ομαλή λειτουργία των πληροφοριακών συστημάτων που χρησιμοποιούν, γίνεται όλο και περισσότερο εμφανές πόσο αναγκαία είναι η ασφάλεια των συστημάτων αυτών, όχι μόνο για την ομαλή λειτουργία του ίδιου του συστήματος αλλά και για την διαφύλαξη των δεδομένων τους. Όπως τα φυσικά κτίρια έχουν κάποια μορφή ασφάλειας σε περίπτωση κάποιας απειλής, έτσι και για τα πληροφοριακά συστήματα, έχουν αναπτυχθεί τρόποι ώστε να μπορέσουν να απαντηθούν ή ακόμα και να προβλεφθούν τέτοιες απειλές. Ο εντοπισμός εισβολών και η ομαλή τους εξουδετέρωση είναι υψίστης σημασίας οπότε ο ρόλος των συστημάτων αυτών είναι ζωτικός στα σύγχρονα συστήματα. Το 1980 ο James Anderson ήταν από τους πρώτους που δημοσίευσαν μια ιδέα [1] για ένα σύστημα που να μπορεί να βοηθήσει τους υπεύθυνους ασφαλείας του δικτύου στην ανασκόπηση των αυτοματοποιημένων από το σύστημα εγγράφων ελέγχου (audit trails ή audit logs). Ο σκοπός του ήταν μέσα από αυτά τα διαθέσιμα logs, όπως για παράδειγμα τις ώρες που συνδέεται (log in) ο χρήστης στο σύστημα σε βάθος ενός μήνα, το πόσες φορές αλληλεπιδρά με ένα αρχείο σε συγκεκριμένο χρόνο ή την υπολογιστική ισχύ που χρησιμοποίησε να μπορέσει να υπολογίσει την πιθανότητα κακόβουλης ενέργειας ή επίθεσης στο σύστημα. Λίγα χρόνια μετά, η Dorothy E. Denning στη δημοσίευση της “An Intrusion-Detection Model” [2] παρουσίασε την μεθολογία που έθεσε τις βάσεις που ενέπνευσαν πολλές έρευνες και υλοποιήσεις που ήταν απαραίτητες για την αναπτυχθεί η τεχνολογία σε αυτό που ονομάζουμε σήμερα σύστημα ανίχνευσης εισβολών ή αλλιώς Intrusion Detection Systems (IDS).

1.1.1 Γιατί είναι απαραίτητα τα συστήματα IDS

Με την ραγδαία αύξηση της δημοφιλίας των υπολογιστικών συστημάτων στην ζωή μας επέρχεται και η εξάρτηση των ανθρώπων σε αυτούς για την ορθή λειτουργία της προσωπικής τους ζωής αλλά και των συστημάτων που υποστηρίζουν τις διάφορες δομές τους. Ανεξαρτήτως μεγέθους ή τομέα ολοένα και περισσότερες εταιρίες, οργανισμοί αλλά και το ευρύ κοινό βλέπουν πλεονεκτήματα στην υιοθέτηση υπολογιστικών συστημάτων, καθημερινά αλληλεπιδρούμε με όλο και περισσότερους υπολογιστές ακόμα και για βασικά πράγματα όπως η πληρωμή αγαθών (τραπεζική κάρτα) ή αγορά εισιτηρίων για τα μέσα μαζικής μεταφοράς (ATH.ENA Card) και εξαρτόμαστε όλο και περισσότερο από την ομαλή λειτουργία τους. Ενδεικτικά το 2019 το 47.1% των νοικοκυριών παγκοσμίως είχαν υπολογιστή στο σπίτι [3], ενώ το ποσοστό στις ανεπτυγμένες χώρες άγγιζε το 80% [4]. Η ίδια αύξηση σημειώνεται και στον εργασιακό τομέα, με το 58% των εργαζόμενων εντός της Ευρωπαϊκής Ένωσης να κάνουν χρήση υπολογιστή με σύνδεση στο διαδίκτυο [5] ενώ ο παγκόσμιος προϋπολογισμός για τον τομέα της τεχνολογίας των πληροφοριών (Information technology - IT) ανέρχεται στα 4,2 δισεκατομμύρια και υπολογίζεται ότι θα φτάσει τα 4,6 δισεκατομμύρια το 2024 [6]. Καθώς η ανθρωπότητα επενδύει και χρησιμοποιεί όλο και περισσότερο τα υπολογιστικά συστήματα, γίνεται απαραίτητη η θωράκιση τους από λάθη ή κυβερνοεπιθέσεις. Το 2017 η επίθεση (τύπου ransomware) WannaCry που διήρκησε μόλις 8 ώρες, κρυπτογραφώντας τα δεδομένα του συστήματος, μόλυνε εκατοντάδες χιλιάδες υπολογιστές σε πάνω από 99 χώρες [7], ζητώντας χρήματα

προκειμένου να αποκρυπτογραφηθούν τα αρχεία. Κόστισε δισεκατομμύρια σε ζημιές επηρεάζοντας, αυτοκινητοβιομηχανίες [8][9], τράπεζες [10], εταιρείες κατασκευής αεροσκαφών [11], πανεπιστήμια [12], κυβερνήσεις κρατών [13] ακόμα και νοσοκομεία [14] βάζοντας σε κίνδυνο ανθρώπινες ζωές.

Όσο περισσότερα υπολογιστικά συστήματα χρησιμοποιούνται, τόσο περισσότεροι στόχοι υπάρχουν για τους επίδοξους επιτιθέμενους. Η γνώση που πρέπει να έχει κάποιος για να την εκτέλεση επιθέσεων όλο και μειώνεται [15], καθώς δημιουργήθηκαν αυτοματοποιημένα εργαλεία (tools ή scripts) που χρειάζονται ελάχιστες γνώσεις από τον χρήστη. Μια επίθεση δεν χρειάζεται να είναι στοχοποιημένη, είναι δυνατή η δημουργία εργαλείου, που συνήθως ονομάζεται bot (απο το robot) και "ψάχνει" όλο το δίκτυο (υπολογιστές, switches, routers, server, κ.α) προσπαθώντας να βρει γνωστά κενά ασφαλείας. Αν επιτύχει μπορεί είτε να ενημερώσει τον επιτιθέμενο, είτε να συνεχίσει το ίδιο την επίθεση.



Σχήμα 1: Πολυπλοκότητα επίθεσης σε σχέση με το γνωστικό επίπεδο επιτιθέμενου [5].

Έτσι σε αντίθεση με παλαιότερα, που ο επιτιθέμενος ήταν άνθρωπος με υψηλές γνώσεις πάνω στο αντικείμενο της ασφαλείας και έπρεπε να αναπτύξει μεμονωμένες μεθόδους και εργαλεία για κάθε επίθεση που επιχειρούσε, σήμερα κάποιος μπορεί απλά να αναζητήσει ένα εργαλείο ή να επισκεφτεί μια ιστοσελίδα και να αγοράσει μια επίθεση, δίνοντας μόνο τα χρήματα και το στόχο της επίθεσης. Μια από τις πιο συνηθισμένες επιθέσεις τέτοιου τύπου είναι η κατανεμημένη επίθεση άρνησης υπηρεσίας (Distributed Denial of Service - DDoS), στις οποίες ένας διακομιστής υπερφορτώνεται με συνδέσεις, λαμβάνει χιλιάδες αιτήματα το δευτερόλεπτο από διαφορετικές διευθύνσεις (Distributed) με σκοπό την εξάντληση των πόρων του. Ο εντοπισμός μιας τέτοιας επίθεσης είναι υπερβολικά δύσκολος, καθώς είναι σχεδόν αδύνατον να διαχωριστούν τα αληθινά αιτήματα από αυτά που είναι μέρος της επίθεσης. Το Microsoft Azure είδε αύξηση 24% στις επιθέσεις τύπου DDoS που έγιναν το 2021 σε σχέση με το προηγούμενο έτος [16]. Η ευκολία στην εκτέλεση και το χαμηλό τεχνικό υπόβαθρο που χρειάζεται για να εκτελεστεί μια τέτοια επίθεση σημαίνει πως γίνονται πιο συχνά και είναι όλο και περισσότερες. Συνεπώς, ο σύνδεσμος της αυξανόμενης πολυπλοκότητας των συστημάτων, των ολέθριων συνεπειών που μπορεί να έχει η διακοπή της λειτουργίας τους, η ευκολία και η ραγδαία αύξηση των επιθέσεων σημαίνουν πως γίνεται αδύνατη η επιτήρηση ενός συστήματος από άνθρωπο, χρειάζεται η ύπαρξη ενός συστήματος που μπορεί να επεξεργαστεί χιλιάδες πληροφορίες το δευτερόλεπτο χωρίς λάθη, ικανό να αναλύσει όλες τις ενέργειες που συμβαίνουν, προκειμένου να υπάρχει σοβαρή πιθανότητα αναγνώρισης ή και καταπολέμησης κάποιας επίθεσης.

1.2 Εισβολές και απειλές

Επιτιθέμενος ή εισβολέας είναι οποιοσδήποτε προσπαθεί να αποκτήσει πρόσβαση σε ένα ασφαλές σύστημα χωρίς να έχει άδεια, για να προκαλέσει σκόπιμα αρνητικό αντίκτυπο στην εμπιστευτικότητα, την ακεραιότητα ή τη διαθεσιμότητα των πληροφοριών, των συστημάτων ή της υποδομής. Οι λόγοι της επίθεσης μπορεί να είναι προσωπικοί, όπως για παράδειγμα εκδίκηση, χρηματικό κέρδος, κατασκοπεία ή ακόμα και κοινωνικοπολιτικοί λόγοι γνωστοί και ως hacktivism. Σε κάθε περίπτωση οι παραπάνω ενέργειες εκτός από παράνομες μπορεί να είναι και επικίνδυνες ειδικά όταν είναι αυτοματοποιημένες, αναζητώντας κενά ασφαλείας σε οποιοδήποτε υπολογιστικό σύστημα ή δίκτυο καταφέρουν να προσχωρήσουν ακόμα και αν πρόκειται για συστήματα νοσοκομείων ή πυρηνικών εργοστασίων. Όταν αναφέρεται ωστόσο ο όρος κυβερνοασφάλεια, ιδιαίτερη έμφαση δίνεται στην έννοια της προστασίας από επιθέσεις που προκύπτουν από εξωτερικές πηγές. Παρόλα αυτά, γίνεται ολοένα και εμφανέστερο, το γεγονός ότι η μεγάλη απειλή για την ασφάλεια ενός οργανισμού μπορεί κάλλιστα να βρίσκεται και μέσα σε αυτόν. Πρόσφατες μελέτες [17] υποδεικνύουν πως το 53% των αναφερόμενων περιστατικών ασφάλειας, οφείλονται σε απειλές ή λάθη προερχόμενες από το δίκτυο του ίδιου του οργανισμού. Οι ανάγκες για συνεργασία και χρήση κοινών πληροφορίων μεταξύ των διαφορετικών γραφείων μιας εταιρίας ή συνεργαζομένων επιχειρήσεων, καθώς και η άνοδος στη χρήση cloud εφαρμογών κάνουν ολοένα και πιο μεγάλο το νούμερο των δικαιούχων που έχουν προνομιακή πρόσβαση στα δίκτυα, τα συστήματα ή τα δεδομένα ενός οργανισμού.

1.2.1 Τι ονομάζουμε εσωτερικά επιτιθέμενο ή εσωτερική απειλή

Εσωτερικά επιτιθέμενος ή εισβολέας (Inside Intruder/Attacker) ορίζεται κάποιος ο οποίος δημιουργεί πρόβλημα σε ένα σύστημα έχοντας πρόσβαση ή δικαιώματα σε αυτό. Πρόκειται δηλαδή για οποιαδήποτε απειλή προέρχεται από τον ίδιο τον στόχο (εταιρία, οργανισμός, υπολογιστικό σύστημα, κ.α). Υπάρχουν πολλοί τύποι εσωτερικών απειλών, ωστόσο μπορούμε να διακρίνουμε τρείς μεγάλες κατηγορίες:

- Κακόβουλοι χρήστες (inside attacker): Πρόκειται για χρήστες που έχουν νόμιμα δικαιώματα πρόσβασης στο σύστημα, όπως για παράδειγμα εργαζόμενοι (παλαιότεροι ή ακόμα και επί του παρόντος), έμπιστοι συνεργάτες ή αλλά τρίτα έμπιστα μέλη. Αποτελεί μια από τις πιο επικίνδυνες απειλές καθώς γίνεται από κακόβουλο χρήστη που πιθανώς γνωρίζει το σύστημα ή την τοπολογία του δικτύου εκ των προτέρων. Αυτή η γνώση και εξοικείωση του με το σύστημα στο οποίο επιτίθεται τον κάνει ακόμα πιο επικίνδυνο, του επιτρέπει να κινηθεί πιο γρήγορα και αποτελεσματικά χωρίς να χρειάζεται να αντιμετωπίσει όλα τα συστήματα ασφαλείας, καθώς είτε μπορεί να τα απενεργοποιήσει, είτε μπορεί, γνωρίζοντας που βρίσκονται, το τρόπο που λειτουργούν και την παραμετροποίηση τους, να τα αποφύγει και να κινηθεί με τρόπο τέτοιο ώστε να μην καταγραφούν ή να γίνουν αντιληπτές οι ενέργειες του.

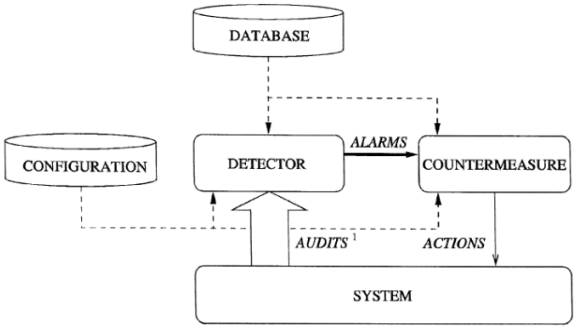
- Απρόσεκτοι χρήστες (careless insider): Ένας απρόσεκτος ή αδιάφορος χρήστης μπορεί εν αγνοία του να προκαλέσει πρόβλημα στο σύστημα, είτε από κάποιο λάθος του, είτε πέφτοντας θύμα απάτης. Η απώλεια εταιρικών συσκευών, στις οποίες συχνά εγκαθίστανται εργαλεία που επιτρέπουν την εύκολη

πρόσβαση στο εσωτερικό δίκτυο (Virtual Private Network - VPN), το άνοιγμα άγνωστων συννημένων που μπορεί να περιέχουν ιούς ή η ψυχολογίκη χειραγώγηση (social engineering), όπως σε επιθέσεις τύπου phishing [18] είναι επιθέσεις που εκμεταλλεύονται τον ανθρώπινο παράγοντα με σκοπό την εύρεση εισόδου (κλεμμένος υπολογιστής, κωδικοί πρόσβασης, κ.α) στο σύστημα.

- Χρήστες προσποιούμενοι νόμιμους χρήστες (imposter): Πρόκειται για εξωτερικά επιτιθέμενους που αποκτούν πρόσβαση στο σύστημα φερόμενοι ως νόμιμοι χρήστες. Συνήθως αποκτούν πρόσβαση σε αυτό επιτιθέμενοι σε απρόσεκτους νόμιμους χρήστες.

1.3 Τι είναι ένα σύστημα ανίχνευσης εισβολών

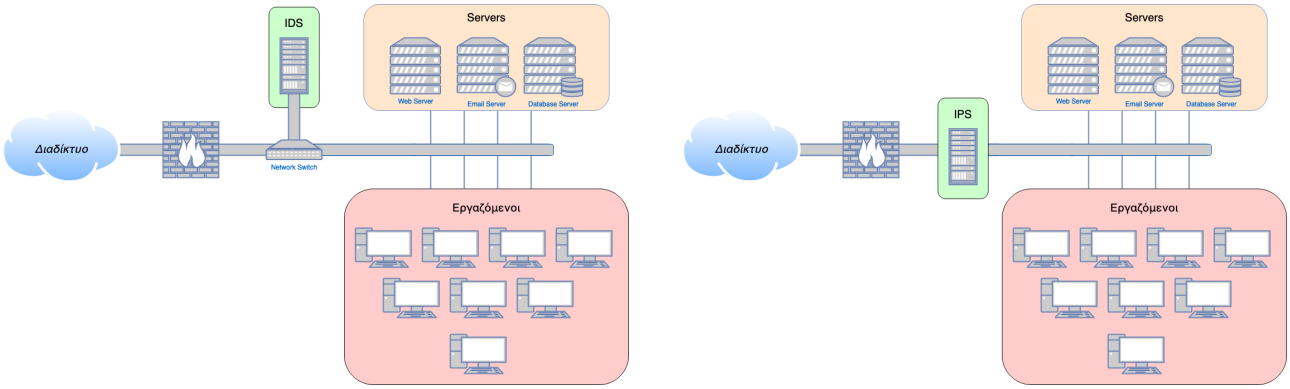
Σύστημα ανίχνευσης ή ιχνηλάτησης εισβολής είναι ένα σύστημα παρακολούθησης και καταγραφής των δραστηριοτήτων που εκτελούνται μέσα σε ένα πληροφοριακό σύστημα ή δίκτυο όπως το άνοιγμα/ μεταφορά/λήψη/τροποποίηση αρχείων, log in, κτλ, με σκοπό των εντοπισμό υπόπτων δραστηριοτήτων, εισβολών ή παραβιάσεων. Μπορούμε να ορίσουμε ένα IDS ως έναν ανιχνευτή (Detector) που αποθηκεύει και επεξεργάζεται τις πληροφορίες που λαμβάνει από το προστατευόμενο σύστημα χρησιμοποιώντας τρεις κατηγορίες πληροφοριών, μακροπρόθεσμες πληροφορίες (Database), όπως για παράδειγμα την λίστα των αρχείων που πρέπει να ελέγχει ή τις θέσεις μνήμης του συστήματος που πρέπει να παραμένουν αμετάκλητες. Επίσης πρέπει να συλλέγει πληροφορίες σχετικά με την τρέχουσα κατάσταση του συστήματος (Configuration) και τέλος πληροφορίες που περιγράφουν τα γεγονότα που συμβαίνουν στο σύστημα (Audits) [19]. Ο σκοπός του Detector είναι να διακρίνει ποιές από τις εισερχόμενες από το σύστημα πληροφορίες είναι σημαντικές, δημιουργώντας έτσι μια βάση (log) με ενέργειες που πραγματοποίησε ο χρήστης. Στη συνέχεια μπορεί να γίνει ανάλυση αυτής της βάσης, αξιολογώντας την επικινδυνότητα των ενεργειών και των αποτελεσμάτων τους προκειμένου να υπολογιστεί η πιθανότητα αυτές να είναι μέρος κάποιας επίθεσης. Προσπαθεί δηλαδή να καταγράψει την εισβολή ή επίθεση, του τύπου της και από ποιον προέρχεται.



Σχήμα 2: Απεικόνιση απλού συστήματος ανίχνευσης [1].
Το πάχος των γραμμών αντιπροσωπεύει τον όγκο της πληροφορίας που μεταφέρεται.

1.3.1 Πρόληψη εισβολών

Μερικά συστήματα έχουν την δυνατότητα επεξεργασίας των καταγεγραμμένων δεδομένων (system activities) επιχειρώντας να αναγνωρίσουν σε πραγματικό ή και δεύτερο χρόνο εισβολές και σε περίπτωση που κρίνουν ότι πρόκειται για επίθεση μπορούν όχι μόνο να προειδοποιούν κάποιον διαχειριστή συστήματος ή αναλυτή του κέντρου επιχειρήσεων ασφαλείας (Security Operations Center - SOC), αλλά και να επιχειρούν να την σταματήσουν, με αυτοματοποιημένο τρόπο χρησιμοποιώντας σωστά αντίμετρα, με ενέργειες όπως η αλλαγή των ρυθμίσεων του firewall ή την φραγή πακέτων απο ύποπτες IP. Ονομάζονται συστήματα ανίχνευσης και πρόληψης εισβολών (Intrusion detection and prevention systems - IDPS ή Intrusion detection systems - IPS) [20]. Σε αντίθεση με τα πιο παραδοσιακά IDS, αυτά συνήθως τοποθετούνται σε σειρά με το σύστημα όπως φαίνεται στο παρακάτω παράδειγμα εφαρμογής ενός IDS και IPS συστήματος στο δίκτυο μιας εταιρίας. Με το τρόπο αυτό, οποιαδήποτε ενέργεια πρέπει πρώτα να εγκριθεί από το σύστημα πριν εκτελεστεί, έχοντας έτσι την δυνατότητα να προλάβουν και να σταματήσουν απειλές πριν καν αυτές ξεκινήσουν.



Σχήμα 3: Διαφορές στην τοπολογία δικτύου ενός συστήματος IDS και IPS

Σήμερα ωστόσο δεν υπάρχει ουσιαστικός διαχωρισμός μεταξύ των παραπάνω όρων καθώς, αν όχι όλα, τα περισσότερα συστήματα πρόληψη εισβολών είναι σε θέση, πέρα από την καταγραφή συμβάντων να πραγματοποιήσουν αυτόματα ενέργειες έτσι ώστε να αναχαιτίσουν μια επίθεση.

1.3.2 Μεθοδολογίες αναγνώρισης απειλών

Αν και υπάρχουν πολλές τεχνικές και μεθοδολογίες για την αναγνώριση επιθέσεων, οι πιο συνηθισμένες που χρησιμοποιούν τα περισσότερα συστήματα είναι οι παρακάτω.

- Συστήματα ανίχνευσης υπογραφής ή Signature-based: Η λειτουργία τους βασίζεται σε μια βάση ψηφιακών υπογραφών γνωστών επιθέσεων. Η ψηφιακή υπογραφή πρόκειται για μια σειρά από bytes που παρατηρούνται μέσα στα μολυσμένα αρχεία ή τα πακέτα που λαμβάνονται από το δίκτυο [21].

Σχήμα 4 :Ακολουθία byte που μπορεί να αποτελεί ψηφιακή υπογραφή [20].

Ωστόσο για τις σύγχρονες επιθέσεις κάτι τέτοιο δεν αρκεί, έτσι διατηρούνται στοιχεία (Indicators Of Compromise - IOC) όπως το αποτέλεσμα κατακερματισμού αρχείων, γνωστές κακόβουλες διευθύνσεις (ιστοσελίδες, IP, κ.α) ακόμα και τα περιεχόμενα ενός εισερχομένου e-mail. Επειδή βασίζονται στη σύγκριση αυτών των υπογραφών και στοιχείων με την βάση που διατηρούν η συχνή ενημέρωση της είναι άκρως σημαντική. Αν και χρειάζεται ελάχιστη επεξεργαστική δύναμη, τα μειονεκτήματα αυτού του τρόπου αναγνώρισης εισβολών είναι πως υστερούν στην αναγνώριση καινούργιων, μη ακόμα καταχωρημένων επιθέσεων.

- Συστήματα ανίχνευσης ανωμαλιών ή Anomaly-based: Χρησιμοποιώντας στατιστικά στοιχεία ή ακόμα και νευρωνικά δίκτυα προσπαθώντας να ανιχνεύσουν άγνωστες επιθέσεις συγκρίνοντας τις ενέργειες του συστήματος με αυτό που θεωρούν συνηθισμένο (baseline) και υπολογίζοντας πόσο παρεκκλίνουν. Συνήθως χρειάζονται χρόνο για να εκπαιδευτούν και να συλλεχθούν αρκετά στοιχεία ώστε να μάθουν το τρόπο λειτουργίας του συστήματος ή του χρήστη για να είναι σε θέση να αναγνωρίζουν κακόβουλες αλληλουχίες ενεργειών, εντοπίζοντας τυχόν ανώμαλες δραστηριότητες. Όπως για παράδειγμα υπερβολική χρήση πόρων, σύνδεση του χρήστη (log on) σε ασυνήθιστη ώρα ή αυξημένος αριθμός συνδέσεων στη βάση μέσα στη μέρα. Ανάλογα με τον τρόπο λειτουργίας και εκπαίδευσης ενδέχεται να χρειάζονται μεγάλη επεξεργαστική δύναμη και χρόνο. Ενώ ειδικά στην περίπτωση των νευρωνικών δικτύων είναι απαραίτητη η επανεκπαίδευση του συστήματος ανά χρονικά διαστήματα με περισσότερα και πιο πρόσφατα δεδομένα (Dataset), μια διαδικασία που απαιτεί πολλούς πόρους.

1.3.3 Κατηγορίες συστημάτων ανίχνευσης

Καθώς αυξήθηκαν οι ανάγκες και οι απαιτήσεις από τα υπολογιστικά συστήματα αυξήθηκε και η πολυπλοκότητα τους προκειμένου να τις υποστηρίξουν. Έτσι υπήρξε η ανάγκη να αναπτυχθούν πολλά συστήματα ανίχνευσης εισβολών, καθένα από αυτά επιβλέποντας διαφορετικά σημεία του συστήματος, προκειμένου τον καλύτερο έλεγχο κάθε μέρους. Μπορούμε να διακρίνουμε δυο μεγάλες κατηγορίες συστημάτων ανίχνευσης εισβολών, τα συστήματα ανίχνευσης εισβολής υπολογιστή (HIDS - Host-based Intrusion Detection System) και τα συστήματα ανίχνευσης εισβολής δικτύου (NIDS - Network Intrusion Detection System) [22].

- HIDS: Πρόκειται για τα πρώτα συστήματα ανίχνευσης εισβολών που δημιουργήθηκαν. Ο σκοπός τους είναι η παρακολούθηση και ανάλυση των συστημάτων στα οποία εγκαθίστανται. Έχουν την δυνατότητα να παρακολουθούν τα εισερχόμενα και εξερχόμενα πακέτα (network packets) του συστήματος που επιτηρούν, την κατάσταση του συστήματος, την χρήση πόρων των προγραμμάτων που τρέχουν, τις αλλαγές στη μνήμη (RAM) και τον αποθηκευτικό χώρο. Τα μειονεκτήματα αυτής της υλοποίησης είναι πως δεν μπορεί να καταγράψει κινήσεις στο δίκτυο (επίθεση σε πολλαπλούς υπολογιστές) και ότι πρέπει να εγκατασταθεί σε κάθε σύστημα ενός δικτύου ξεχωριστά, δηλαδή τοπικά. Πέρα από την δυσκολότερη διαχείρηση του συστήματος και το πιθανό κόστος στις επίδοσης του, καθώς χρησιμοποιεί κάποιους πόρους για την λειτουργία του, αυτό σημαίνει ότι συνήθως αποτελεί πρωταρχικό στόχο ενός επιτιθέμενου που αποκτά πρόσβαση στο σύστημα. Θα μπορούσαμε να πούμε ότι ένα παράδειγμα ενός σύγχρονου HIDS είναι τα προγράμματα antivirus τα οποία εγκαθίστανται τοπικά και ελέγχουν και προστατεύουν μόνο το τοπικό αυτό μηχάνημα.

- NIDS: Εγκαθίστανται σε ένα ή περισσότερα κομβικά σημεία ενός δικτύου, με σκοπό να ελέγχουν όλα τα πακέτα που ανταλλάσσονται μέσα σε αυτό, προστατεύοντας το δίκτυο, και κατά συνέπεια όλες τις συσκευές που το απαρτίζουν όπως υπολογιστές, διακομιστές (servers), δρομολογητές (routers), κ.α, χωρίς να απαιτούνται πόροι από τα συστήματα αυτά. Χαρακτηρίζονται από την ευκολότερη εγκατάσταση (καθώς δεν χρειάζεται να υπάρχει ένα IDS για κάθε μηχάνημα) και διαχείρηση. Το μειονέκτημα του συγκεκριμένου τύπου IDS, είναι πως μπορεί να δημιουργήσουν καθυστερήσεις στο δίκτυο (bottleneck).

Για τις δυο παραπάνω κατηγορίες υπάρχουν και οι παραλλαγές HIPS (Host-based Intrusion Prevention System) και NIPS (Network Prevention Detection System), εξελίξεις της τεχνολογίας που επιτρέπουν πέρα από την αποστολή ειδοποιήσεων, την αυτόματη εκτέλεση ενεργειών ή αλλαγή ρυθμίσεων ώστε να σταματήσουν την επίθεση.

1.3.4 Αποτελεσματικότητα ενός συστήματος ανίχνευσης

Η αποτελεσματικότητα ενός IDS είναι άκρως σημαντική, ένα σύστημα που δεν καταφέρνει να αναγνωρίσει επιθέσεις και ορίζει ως επικίνδυνες, επιτρεπτές ενέργειες (false positive) ή ακόμα χειρότερα το αντίθετο όχι μόνο δημιουργεί σύγχυση στους χρήστες, αλλά και προσφέρει λίγα για την ασφάλεια του υπολογιστικού συστήματος ή του δικτύου. Η αποτελεσματικότητα της υλοποίησης βασίζεται στις παρακάτω μεταβλητές [1].

- Ακρίβεια (accuracy): Το σύστημα να μην ορίζει σαν επικίνδυνες (flag), νόμιμες ή αποδεκτές ενέργειες ή να μην κάνει flag παράνομες ενέργειες μέσα στο σύστημα

- Επίδοση και ταχύτητα (Performance): Το ποσό γρήγορα μπορεί να επεξεργαστεί τις εισερχόμενες πληροφορίες καθώς αν υπάρχουν καθυστερήσεις, δεν υπάρχει δυνατότητα ανίχνευσης των επιθέσεων σε πραγματικό χρόνο.

- Πληρότητα (Completeness): Η δυνατότητα να ανιχνεύει όλες τις επιθέσεις στο σύστημα, κάτι που είναι δύσκολο να υπολογιστεί καθώς δεν είναι δυνατόν να υπάρχει βάση με όλες τις επιθέσεις, αν αυτές δεν έχουν πρώτα ανιχνευτεί από κάποιον.

- Ανοχή σε σφάλματα (Fault tolerance): Ένα τέτοιο σύστημα είναι σημαντικό να είναι το ίδιο ανθεκτικό σε επιθέσεις. Διαφορετικά μπορεί ο επιτιθέμενος να το στοχοποιήσει πριν την επίθεση του, προκειμένου να κινηθεί ανενόχλητος ή μετά το πέρας της επίθεσης, για να διαγράψει τα ενοχοποιητικά στοιχεία.

- Χρόνος απόκρισης (Timeliness): Για συστήματα IDPS, είναι ο χρόνος που περνάει από την στιγμή που γίνεται η επίθεση, μέχρι να ληφθούν ειδοποιήσεις από το σύστημα. Αργοπορίες στην επεξεργασία και αναγνώριση των επιθέσεων, δυσχεραίνουν τις προσπάθειες αναχαιτίσης της επίθεσης από τους υπεύθυνους ή το σύστημα

Όλα τα παραπάνω εξαρτώνται από τη αξιόπιστη συλλογή και διατήρηση απροσπέλαστων στοιχείων τα οποία είναι κρίσιμα για την επιτυχημένη λειτουργία ενός IDS. Αυτά είναι που χρησιμοποιεί τόσο για την ανάλυση της επίθεσης σε πραγματικό χρόνο, όσο και για την μελέτη της επίθεσης μετά το πέρας της. Ένα σύστημα που συλλέγει στοιχεία τα οποία μπορούν εύκολα να τροποποιηθούν ή να διαγραφούν δεν μπορεί να φέρει σε πέρας επιτυχημένα το σκοπό του.

1.4 Γιατί ένας εσωτερικά επιτιθέμενος βρίσκεται σε πλεονεκτική θέση

Αν και οι επιθέσεις εκ των έσω είναι μια από τις πιο επικίνδυνες απειλές για ένα σύστημα, είναι πολύ δύσκολο να προβλεφθούν, να εντοπιστούν και να αμυνθούν λόγω της εμπιστοσύνης και των ευθυνών που έχουν οι εργαζόμενοι [23], ενώ τα παραδοσιακά αντίμετρα (IDS), δεν είναι πάντα σε θέση να προστατεύσουν από μια τέτοια απειλή. Σε αντίθεση με μια εξωτερική απειλή που πρέπει να ξεπεράσει πολλαπλά στάδια ασφαλείας (security layers), ένας εσωτερικά επιτιθέμενος ίσως να χρειαστεί να διαπεράσει πολύ λιγότερα ή και κανένα μόνο και μόνο λόγω των δικαιωμάτων που κατέχει. Επίσης ακόμα και στην περίπτωση που υπάρχουν συστήματα ασφαλείας για την επιτήρηση τοπικών ενεργειών, η γνώση ή τα δικαιώματα του μπορεί να του επιτρέπουν να τα απενεργοποιήσει.

Εξωτερικός κόσμος	
1η άμυνα	IDS 1
2η άμυνα	IDS 2
...	...
v-1 άμυνα	IDS v-1
v άμυνα	IDS v
Δεδομένα συστήματος	

Σχήμα 5: Παράδειγμα συστημάτων ασφαλείας που πρέπει να ξεπεράσει μια εξωτερική απειλή

Όπως παρατηρείται παραπάνω στο σχεδιάγραμμα (Σχήμα 5) υπάρχει ο εξωτερικός κόσμος και ύστερα τα εκάστοτε επίπεδα άμυνας (defense layers), συνοδευόμενα και από τα αντίστοιχα συστήματα ανίχνευσης, στο τέλος των οποίων, βρίσκεται το εσωτερικό σύστημα (για παράδειγμα μια βάση δεδομένων) που είναι το κίνητρο για την εισβολή. Για κάθε επίπεδο υπάρχει και ένα αντίστοιχο σύστημα ανίχνευσης εισβολής, προσαρμοσμένο για της ανάγκες του επιπέδου (υπολογιστής, server, network switch κ.α). Ένας εξωτερικός εισβολέας, θα πρέπει να είναι ικανός να ξεπεράσει κάθε ένα από τα N επίπεδα ασφαλείας καθώς και να παρακάμψει όλα τα συστήματα ανίχνευσης που τα συνοδεύουν. Ειδικά σε περιπτώσεις πολλών τέτοιων στρωμάτων, η επίτευξη της εισβολής καθίσταται πολύ δύσκολη. Ένας εσωτερικά επιτιθέμενος βρίσκεται ήδη μέσα στο σύστημα, έχει ήδη ξεπεράσει δηλαδή έναν αριθμό από επίπεδα άμυνας, αυξάνοντας την πιθανότητα επιτυχίας της επίθεσης. Τέλος, σε μια επίθεση τέτοιου τύπου υπάρχουν μεγαλύτερες πιθανότητες να διαγραφούν τα δεδομένα που συλλέγει το σύστημα, κάνοντας την αναγνώριση και ανάλυση της επίθεσης ακόμα πιο δύσκολη. Με τη διαγραφή των ενοχοποιητικών στοιχείων γίνεται αδύνατη η πρόσαψη ευθυνών και νομική κίνηση εναντίον του επιτιθέμενου γίνεται ακόμα πιο δύσκολη.

1.5 Περιστατικά εσωτερικών εισβολών

1.5.1 Περίπτωση της Fannie Mae

Η απόπειρα επίθεσης από έναν υπάλληλο της Fannie Mae μετά την απόλυση του είναι ένα τέλειο παράδειγμα μιας εσωτερικής απειλής που πιθανώς υποκινείται από εκδίκηση [24]. Πιο συγκεκριμένα, στη Βαλτιμόρη, ο προγραμματιστής Rajendrasinh Babubhai Makwana καταδικάστηκε σε 41 μήνες φυλακή (περίπου 3,5 χρόνια), ακολουθούμενα από 3 χρόνια ελευθερίας υπό παρακολούθηση, λόγω επίθεσης στην εταιρία Fannie Mae. Ο Makwana απολύθηκε από την εταιρία στις 24 Οκτωβρίου του 2008. 5 μέρες αργότερα, στις 29 Οκτωβρίου ένας Senior προγραμματιστής της εταιρίας, βρήκε ένα script κακόβουλου περιεχομένου μέσα στα αρχεία της εταιρείας, το οποίο είχε δρομολογηθεί για να λειτουργήσει στις 31 Ιανουαρίου του 2009. Το script αυτό είχε εισαχθεί στο σύστημα, τη μέρα που απολύθηκε ο Makwana και είχε ως σκοπό να διαγράψει όλα τα δεδομένα που υπήρχαν αποθηκευμένα στους server, συμπεριλαμβανομένων πληροφοριών σχετικά με τα οικονομικά, ασφαλιστικά και αποθηκευμένα στοιχεία της εταιρείας. Ήταν ένα συμβάν ρήξης της εμπιστοσύνης κάποιου που είχε εξειδικευμένη πρόσβαση στα συστήματα της εταιρείας.

1.5.2 Η περίπτωση της δικηγορικής εταιρείας Elliot Greenleaf

Η περίπτωση της δικηγορικής εταιρείας Elliot Greenleaf [25] είναι ένα πολύ καλό παράδειγμα εσωτερικής επίθεσης υποκινούμενης από οικονομικά οφέλη. Πιο συγκεκριμένα, τον Ιανουάριο του 2021, έκανε την εμφάνιση της μία υπόθεση όπου τέσσερις (4) δικηγόροι που εργάζονταν για την εν λόγω εταιρεία έκλεβαν και διέγραφαν επί τέσσερις (4) μήνες email, φακέλους και αρχεία της εταιρείας, με σκοπό να βοηθήσουν ανταγωνιστή τους να ανοίξει γραφείο στην περιοχή. Από τα τέλη Οκτώβρη του 2020 και μέχρι τον Ιανουάριο του 2021, δρούσαν δίχως άδεια με τα αρχεία της εταιρίας, εκθέτοντας στοιχεία απόρρητα με αποτέλεσμα το γραφείο στο οποίο εργάζονταν, ύστερα από την αποχώρηση τους, να κλείσει. Το πρόβλημα έγινε αντιληπτό επειδή η εταιρία κρατούσε εφεδρικά αντίγραφα ηλεκτρονικού ταχυδρομείου. Σε αντίθεση με την προηγούμενη περίπτωση της Fannie Mae όπου ο δράστης ήταν Senior Programmer στην εταιρεία και είχε πρόσβαση σε πολύ πιο τεχνικά κομμάτια της εταιρείας του, σε αυτή την περίπτωση, οι δράστες ήταν δικηγόροι, δηλαδή άνθρωποι με μικρή σχετικά τεχνολογική κατάρτιση που όμως είχαν, αυξημένα δικαιώματα στο σύστημα (διαγραφή δεδομένων).

1.5.3 Η περίπτωση του Αστυνομικού Τμήματος του Dallas

Η περίπτωση της απώλειας δεδομένων του αστυνομικού τμήματος του Dallas στην Αμερική [26], αποτελεί ένα παράδειγμα όχι τόσο κακόβουλης εσωτερικής απειλής, όσο περίπτωση όπου η απροσεξία και η ελλιπής εκπαίδευση των εργαζομένων, σε συνεργασία με την έλλειψη αποτρεπτικών μέσων, μπορούν να οδηγήσουν σε απώλεια δεδομένων. Ένας νεαρός τεχνικός του οποίου τα στοιχεία δεν έχουν δημοσιευθεί, ανάμεσα στον Μάρτιο και Απρίλιο του 2021, κατέστρεψε πάνω από 8,7 εκατομμύρια αρχεία αστυνομικών υποθέσεων τα οποία περιείχαν βίντεο, φωτογραφίες, ηχογραφήσεις και άλλα στοιχεία. Αποδείχθηκε ότι η πρόθεση του δεν ήταν κακόβουλη. Στην συγκεκριμένη περίπτωση, η παραπάνω εκπαίδευση των

εργαζομένων πάνω στην διαχείριση ευαίσθητων αρχείων είναι απαραίτητη (κυρίως για όσα αφορούν κυβερνητικές υποθέσεις), αλλά επίσης, τα συστήματα ανίχνευσης εισβολών, θα μπορούσαν να είχαν χρησιμοποιηθεί ώστε να προληφθεί μία τέτοια κατάσταση.

1.5.4 Η περίπτωση του Edward Snowden

Η περίπτωση του Edward Snowden είναι ίσως η πιο γνωστή περίπτωση εσωτερικής εισβολής [27] μέχρι σήμερα. Ήταν ο διαχειριστής των συστημάτων της υπηρεσίας Εθνικής Ασφάλειας (NSA) της κυβέρνησης των Ηνωμένων Πολιτειών Αμερικής (ΗΠΑ). Έκλεψε και μοιράστηκε με τον Τύπο εκατομμύρια απόρρητα έγγραφα, τα οποία ύστερα δημοσιεύτηκαν. Επειδή ήταν διαχειριστής συστήματος (προνομιούχος και ο απόλυτος εσωτερικός χρήστης) είχε ευρεία πρόσβαση σε πολλά απόρρητα συστήματα της NSA και στα δεδομένα τους και ήταν σε θέση να αποκτά και να διανέμει κρυφά τεράστιες ποσότητες εξαιρετικά ευαίσθητων δεδομένων. Όμως από τα αρχεία που διέρρευσαν συμπεραίνεται πως είχε πρόσβαση σε συστήματα που δεν θα έπρεπε. Δεν υπήρχαν μόνο απόρρητα έγγραφα του υποδικτύου που διαχειριζόταν ο ίδιος, αλλά όλου του δικτύου της NSA, άλλων συνεργαζόμενων ομοσπονδιακών αρχών και ακόμη και έγγραφα υπηρεσιών συνεργαζόμενων χωρών [28]. Αν και είναι λίγες οι λεπτομέρειες για τον τρόπο της επίθεσης, εικάζεται από αναλυτές πως κατάφερε με τις γνώσεις και τα δικαιώματα που είχε στο σύστημα να μάθει που είναι αποθηκευμένα τα αρχεία που τον ενδιέφεραν, στη συνέχεια έπεισε περίπου δώδεκα συνεργάτες του να του παραχωρήσουν τους κωδικούς τους λέγοντας πως τους χρειάζεται για την δουλειά του, αυτοί των πίστεψαν καθώς γνώριζαν ότι ήταν ο υπεύθυνος δικτύου. Έτσι απέκτησε πρόσβαση σε συστήματα που δεν έπρεπε να έχει. Στη συνέχεια, κρυπτογραφούσε τα αποκτηθέντα αρχεία προκειμένου να παραμένουν κρυμμένα κατά τη μεταφορά τους. Τέλος, πιθανότατα τροποποίησε τα αρχεία που διατηρούσε το σύστημα (log files) προκειμένου να μην αποκαλυφθεί. Αν και οι απόψεις για το πως ακριβώς έγινε η επίθεση διίστανται, οι περισσότεροι συμφωνούν πως αυτό που έκανε δεν θα ήταν δυνατόν αν δεν ήταν στη θέση που ήταν, με τα δικαιώματα που είχε. Είναι από τις ελάχιστες περιπτώσεις γενικότερα που μπορεί να ειπωθεί πως υπήρχαν συστήματα ανίχνευσης και προστασίας τα οποία λειτουργούσαν αλλά παρακάμφθηκαν.

1.6 IDS και Εγκληματολογία Υπολογιστών (Computer Forensics)

Η χρήση ενός προσωπικού υπολογιστή με διάφορους τρόπους, είναι μία περιήγηση σε έναν ψηφιακό κόσμο, όπου όπως και στον πραγματικό, εγκλήματα μπορούν να συμβούν. Ο όρος ηλεκτρονικό έγκλημα ακούγεται ολοένα και συχνότερα και αυτό σε συνάρτηση με την αυξημένη χρήση των υπολογιστών δημιούργησε έναν νέο κλάδο στην επιστήμη της πληροφορικής, την εγκληματολογική (ιατροδικαστική) πληροφορική. Η ιατροδικαστική πληροφορική είναι η επιστήμη της απόκτησης, διατήρησης, ανάκτησης και παρουσίασης δεδομένων ενός συστήματος διασφαλίζοντας πως αυτά δεν τροποποιήθηκαν, προκειμένου την ανάλυση κάποιου συμβάντος. Ήδη από το 1984, το Εργαστήριο του FBI και άλλες υπηρεσίες επιβολής του νόμου άρχισαν να αναπτύσσουν προγράμματα για την εξέταση στοιχείων υπολογιστών. Για αυτό το λόγω γίνεται αντιληπτό το ποσό σημαντικό είναι τα δεδομένα που συγκρατεί ένα σύστημα IDS να είναι απροσπέλαστα ώστε να είναι δυνατή η έρευνα. Ακόμη και αν δεν καταφέρει το σύστημα να προειδοποιήσει

ή να σταματήσει κάποια απειλή, αν διασφαλιστούν τα logs που αποθηκεύτηκαν κατα το συμβάν, μπορούν να αναλυθούν, αποκαλύπτοντας το τρόπο της επίθεσης και επιρρίπτοντας ευθύνες προς τα σωστά άτομα ή συστήματα. Έτσι στη συνέχεια, με αυτά τα δεδομένα γίνεται εφικτή η θωράκιση του συστήματος από αναγνωρισμένες αδυναμίες (αλλαγή κανόνων ασφαλείας) και η μελλοντική καταπολέμηση κακόβουλων δραστηριοτήτων

1.7 Τα προβλήματα των IDS

Στις παραπάνω ενότητες, έγινε μία προσπάθεια ανάδειξης της σημαντικότητας των συστημάτων ανίχνευσης εισβολών και πως αυτά μπορούν να συμβάλλουν καθοριστικά στην αποτροπή δυσάρεστων συμβάντων με ανυπολόγιστο πολλές φορές κόστος. Όμως, όσο και αν η τεχνολογία προοδεύει και νέοι μέθοδοι για εισβολή αλλά και άμυνα εφευρίσκονται, τα IDS θα συνεχίσουν να είναι ατελή. Ο κύριος λόγος για τον οποίο συμβαίνει αυτό είναι διότι οι μηχανές, δεν έχουν νοημοσύνη και τη δυνατότητα κριτικής ανάλυσης. Πώς μπορεί να οριστεί το σβήσιμο ενός email ως κακόβουλη ενέργεια; Το άδειασμα του κάδου ανακύκλωσης αποτελεί εβδομαδιαία σχεδόν διαδικασία για εκατομμύρια χρήστες. Η προσθήκη ή διαγραφή καταχωρίσεων σε βάσεις δεδομένων είναι σύνηθες φαινόμενο. Το να πραγματοποιήσει ένας διαχειριστής ή ένας τεχνικός κάποια αλλαγή στον webserver της εταιρίας του, ενδέχεται να είναι καθημερινή δουλειά. Όλες αυτές οι ενέργειες, είναι απόλυτα φυσιολογικές σε ένα χώρο εργασίας, αλλά μπορούν κάλλιστα να αποτελέσουν τρόπους επίθεσης σε μία εταιρεία ή έναν οργανισμό. Στα παραδείγματα που δόθηκαν παραπάνω, συναντήθηκαν πολλές περιπτώσεις όπου οι θύτες κατά κύριο λόγω εντοπίστηκαν, όχι τόσο λόγω του συστήματος ανίχνευσης, αλλά περισσότερο μετά από τυχαία διερεύνηση από τους συνεργάτες τους (όπως στην περίπτωση της Fannie Mae) ή επειδή δημιούργησαν υποψίες με την περίεργη συμπεριφορά τους (όπως στην περίπτωση της Elliot Greenleaf). Σε κανένα από αυτά τα περιστατικά δεν ήχησαν ειδοποιήσεις για ανώμαλη συμπεριφορά στο σύστημα και αυτή είναι η δυσκολία που καλούνται να αντιμετωπίσουν τα IDS συστήματα. Χωρίς την κριτική ανάλυση των γεγονότων, κάποια διαδικασία δεν μπορεί απαραίτητα να χαρακτηριστεί ως επικίνδυνη. Ακόμα και η πλήρης διαγραφή ενός server μπορεί να αποτελεί μέρος μιας συντήρησης που διεξάγεται από τους τεχνικούς υπολογιστών μιας εταιρίας. Αυτή η αδυναμία διάκρισης λοιπόν του τι μπορεί να θεωρηθεί κακόβουλη, δυνητικά κακόβουλη και άκακη συμπεριφορά είναι το μεγάλο πρόβλημα των συστημάτων αυτών.

Κεφάλαιο 2. Το Blockchain

2.1 Το Blockchain πριν το Blockchain

Το 1991, εκδόθηκε από τους Stuart Haber και W. Scott Stornetta ένα άρθρο [29], το οποίο αποτέλεσε την πρωταρχική ιδέα για αυτό που σήμερα ονομάζεται Blockchain. Σε εκείνο το άρθρο, οι συγγραφείς, πρότειναν μία διαδικασία χρονοσήμανσης (δηλαδή σήμανσης του χρόνου της δημιουργίας ενός αρχείου), η οποία θα ήταν υπεύθυνη για την ψηφιακή επικύρωση του χρόνου στον οποίο το αρχείο δημιουργήθηκε και να επικυρώνει ύστερα τον χρόνο που τροποποιούταν. Χαρακτηριστικά στο άρθρο τους αναφέρουν:

«*Η προοπτική ενός κόσμου στον οποίο όλα τα έγγραφα κειμένου, ήχου, εικόνας και βίντεο είναι σε ψηφιακή μορφή σε εύκολα τροποποιήσιμα μέσα, εγείρει το ζήτημα του τρόπου πιστοποίησης του πότε δημιουργήθηκε ή τροποποιήθηκε τελευταία φορά ένα έγγραφο. Το πρόβλημα είναι η χρονική σήμανση των δεδομένων, όχι του μέσου. Προτείνουμε υπολογιστικά πρακτικές διαδικασίες για ψηφιακή χρονοσήμανση τέτοιων εγγράφων, έτσι ώστε να είναι αδύνατο για έναν χρήστη είτε να θέσει προγενέστερη είτε να θέσει μεταγενέστερη ημερομηνία στο έγγραφο του, ακόμη και μέσω συνεννόησης με κάποια υπηρεσία χρονοσήμανσης. Οι διαδικασίες μας διατηρούν το απόρρητο των ίδιων των εγγράφων και δεν απαιτούν τήρηση αρχείων από την υπηρεσία χρονοσήμανσης.*»

Εκείνη την εποχή, το πρόβλημα που έχριζε λύσης ήταν το γεγονός ότι η πιστοποίηση της ημερομηνίας όπου ένα έγγραφο δημιουργήθηκε και ύστερα τροποποιήθηκε, ήταν σχεδόν ανύπαρκτη και δύσκολη. Από αυτό το πρόβλημα προέκυπταν θέματα απόδειξης της πνευματικής ιδιοκτησίας, όπου ο δημιουργός θα έπρεπε να είναι σε θέση να αποδείξει ότι η πατέντα που είχε δημιουργήσει ήταν πραγματικά προγενέστερη από τις υπόλοιπες, με τις οποίες ερχόταν σε αντιπαράθεση. Μέχρι τότε οι διαδεδομένοι τρόποι τέτοιας επικύρωσης αποτελούσαν είτε την διαχείριση αρχείου εργασιών μέσα στο τετράδιο του εργαστηρίου, όπου υπήρχε καθημερινή καταγραφή της πορείας της έρευνας του, είτε η προσωπική αποστολή μέσω ταχυδρομείου ενός αντιγράφου της έρευνας, το οποίο παρέμενε σφραγισμένο ώστε να αποτελεί απόδειξη ότι ήταν προγενέστερο της ημερομηνίας που στάλθηκε. Οι διαδικασίες αυτές, πέρα από απαιτητικές, είχαν πρακτική ισχύ μόνο στα φυσικά έγγραφα. Η εισαγωγή όμως ψηφιακών εγγράφων, κατέστησε τη διαδικασία αυτή ακόμα πιο περίπλοκη. Πλέον η διαδικασία θα έπρεπε να λαμβάνει υπόψιν της δύο απαιτούμενα ώστε να επικυρωθεί σωστά η ημερομηνία. Το πρώτο προαπαιτούμενο είναι ότι θα έπρεπε να εξασφαλίσει ότι η αλλαγή του εγγράφου ήταν αδύνατη αν δεν ήταν εμφανής από το μέσο στο οποίο εμφανίζεται. Δηλαδή για να σφραγίσει κάποιος τα δεδομένα, θα έπρεπε η πορεία της επεξεργασίας του εγγράφου, να είναι εμφανής. Το δεύτερο προαπαιτούμενο ήταν ότι θα έπρεπε να εξασφαλιστεί ότι ένα έγγραφο θα ήταν αδύνατο να σφραγιστεί με διαφορετική ημερομηνία και ώρα πέρα από την πραγματική. Στο άρθρο τους προτάθηκαν δύο λύσεις. Αμφότερες περιλάμβαναν τη χρήση μονόδρομων συναρτήσεων κατακερματισμού (hash functions), των οποίων οι έξοδοι υποβάλλονταν σε επεξεργασία αντί των πραγματικών εγγράφων.

2.2 Το Blockchain όπως το γνωρίζουμε σήμερα

Ο Satoshi Nakamoto ήταν ο εμπνευστής της τεχνολογίας που σήμερα είναι γνωστή ως Blockchain. Στην εργασία του με τίτλο «Bitcoin: A Peer-to-Peer Electronic Cash System» [30] έθεσε την μαθηματική βάση του κρυπτονομίσματος bitcoin συνδυάζοντας πολλές ιδέες και τεχνολογίες και αποτέλεσε την πρώτη εφαρμογή ενός blockchain. Μέχρι και σήμερα η πραγματική ταυτότητα του Satoshi παραμένει άγνωστη, με πολλούς να πιστεύουν ότι πρόκειται για ομάδα ερευνητών και όχι ενός και μόνο προσώπου. Το άρθρο δεν υποβλήθηκε, και δεν κρίθηκε ποτέ από ένα παραδοσιακό επιστημονικό περιοδικό. Τον Ιανουάριο του 2009, η κυκλοφορία της εφαρμογής Bitcoin σηματοδότησε την πρώτη εφαρμογή που χρησιμοποιούσε την τεχνολογία Blockchain. Αυτό που δημιούργησε είναι ένα ψηφιακό νόμισμα, που επιτρέπει την πληρωμή κατευθείαν από άτομο σε άτομο, χωρίς να χρειάζεται να επεξεργαστεί η συναλλαγή από κάποιον έμπιστο τρίτο (πχ. Τράπεζα). Το μεγαλύτερο πρόβλημα που προσπάθησε να λύσει ήταν η διπλή δαπάνη (double spending), για το οποίο πρότεινε και υλοποίησε ένα σύστημα όπου οι συναλλαγές περνάνε από μια συνάρτηση κατακερματισμού (proof of work) και στη συνέχεια προστίθενται σε ένα ιστορικό συναλλαγών ή αλυσίδα, που δεν μπορεί να αλλαχτεί παρά μόνο χρησιμοποιώντας μεγάλη υπολογιστική ισχύ και υπολογίζοντας τη συνάρτηση κατακερματισμού ολόκληρης της αλυσίδας από την αρχή.

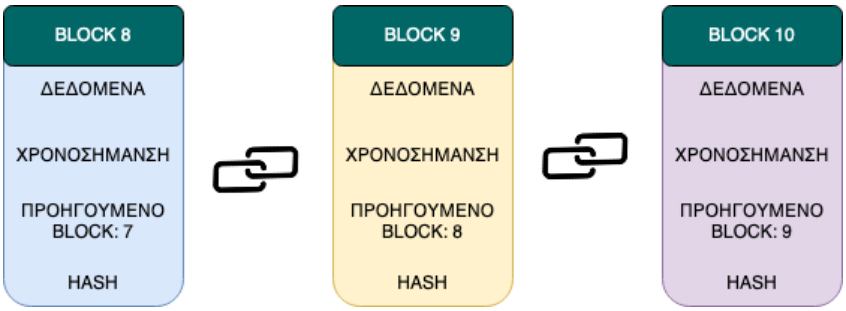
Η τεχνολογία αυτή πέρα του ότι αποτελεί την βάση των κρυπτονομισμάτων, καθώς έκτοτε έχουν δημιουργηθεί πάνω από δώδεκα χιλιάδες διαφορετικά νομίσματα, έχει βρεί ευρεία εφαρμογή και στον πιο παραδοσιακό χρηματοοικονομικό κλάδο (Financial technology - FinTech), κινώντας το ενδιαφέρον τόσο της ευρωπαϊκής κεντρικής τράπεζας η οποία αναπτύσει το ψηφιακό ευρώ [31] που μπορεί να βασιστεί σε τεχνολογίες blockchain όσο και άλλων χρηματοοικονομικών οργανισμών όπως οι Goldman Sachs Group Inc. και Citigroup Inc. που χρησιμοποιούν το Veris ένα σύστημα blockchain που αναπτύχθηκε από την Axoni [32] με σκοπό την καταγραφή των ανταλλαγών μετόχων αλλά και διατήρηση του ιστορικού τους. Σαν ιδέα απέκτησε μεγάλη δημοσιότητα λόγω της αύξησης της αξίας του κρυπτονομίσματος που την δημιούργησε, ωστόσο, μέχρι και σήμερα, 13 χρόνια μετά από την εμφάνισή της, δεν υπήρξαν πολλές εφαρμογές blockchain πέρα από τις δυο προαναφερθείσας κατηγορίες. Σύμφωνα με τον ιστότοπο DappRadar, υπήρχαν 3.047 αποκεντρωμένες εφαρμογές (Decentralised Apps ή DApp), δηλαδή εφαρμογών που χρησιμοποιούν την τεχνολογία blockchain, που βασίστηκαν σε δημοφιλείς δημόσιες αλυσίδες. Μαζί είχαν 69.788 ημερήσιους ενεργούς χρήστες (Daily Active Users - DAU) [33]. Η πλειονότητα τους επικεντρώθηκε σε τομείς όπως τα παιχνίδια, τζόγος και οι ανταλλαγές κρυπτονομισμάτων [34]. Ωστόσο λόγω αυτής της πρωτοφανούς δημοσιότητας έχει προταθεί σαν λύση σε πολλά προβλήματα και ολοένα και ξεφεύγει από αυτούς τους δυο τομείς.

2.3 Πως λειτουργεί το Blockchain

Ένα blockchain πρόκειται για μια κατανεμημένη και δημόσια αλυσίδα μόνο προσαρτωμένων καθολικών, τα δεδομένα της οποίας είναι διαχωρισμένα σε μπλοκ τα οποία συνδέονται μεταξύ τους με χρονολογική σειρά. Προκειμένου όμως να κατανοήσουμε πως ακριβώς λειτουργεί ένα blockchain πρέπει πρώτα να αναλύσουμε και να κατανοήσουμε τις τεχνολογίες που χρησιμοποιεί καθώς ο συνδυασμός όλων αυτών των τεχνολογιών είναι σημαντικός για την λειτουργία του.

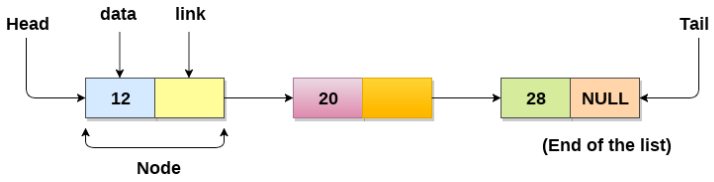
2.3.1 Η αλυσίδα

Η λέξη blockchain αποτελείται από τις λέξεις block, καθώς τα δεδομένα αποθηκεύονται μέσα σε αυτά, και chain, καθώς όλα τα blocks συνδέονται μεταξύ τους και σχηματίζουν μια συνεχόμενη αλυσίδα. Συνεπώς κάθε block πρέπει να περιέχει οπωσδήποτε πέρα από τα δεδομένα προς αποθήκευση, πληροφορίες σχετικά με το αμέσως προηγούμενο στην αλυσίδα block. Συνήθως όμως περιέχει επιπλέον πληροφορίες όπως την ηλεκτρονική χρονοσήμανση ή αλλιώς χρόνο-σφραγίδα δημιουργίας του (timestamp) που ονομάζεται epoch time και πρόκειται για τον αριθμό τον δευτερολέπτων που έχουν μεσολαβήσει από κάποια συγκεκριμένη ημερομηνία που αλλάζει ανά λειτουργικό (πχ. στα UNIX είναι 1η Ιανουαρίου του 1970, στα MS DOS είναι 1η Ιανουαρίου του 1980), μέχρι τη στιγμή δημιουργίας του block και το αποτέλεσμα της συνάρτησης κατακερματισμού (hash) του ιδίου του block, ο σκοπός της οποίας θα αναλυθεί στη συνέχεια.



Σχήμα 6: Απλοποιημένο παράδειγμα blockchain

Έτσι δημιουργείται μια δομή παρόμοια με αυτή που ήδη γνωρίζουμε ως απλά συνδεδεμένες λίστες (single-linked lists) που χρησιμοποιείται για την αποθήκευση στοιχείων διάσπαρτα στη μνήμη ενός υπολογιστή και όχι σε συνεχόμενο χώρο.



Παράδειγμα συνδεδεμένης λίστας.
(Πηγή: www.javatpoint.com/singly-linked-list)

Η μόνη διαφορά είναι, ότι αντί κάθε κόμβος να περιέχει δείκτη προς τον επόμενο κόμβο, περιέχει δείκτη προς τον προηγούμενο. Ο σκοπός αυτής της αλυσίδας είναι η διατήρηση του ιστορικού των δεδομένων και η αποφυγή τροποποίησης αυτού καθώς κάθε block αναφέρει ρητά ποιο είναι το προηγούμενο κάνοντας δυσκολότερο να προστεθεί κάποιο ανάμεσα σε δυο που προϋπάρχουν ή να αλλαχτεί η σειρά τους, διατηρώντας την ακεραιότητα του ιστορικού στην αλυσίδα.

2.3.2 Συναρτήσεις κατακερματισμού

Οι συναρτήσεις κατακερματισμού (hash functions) είναι μαθηματικές συναρτήσεις που δέχονται ως είσοδο δεδομένα και επιστρέφουν ένα ακέραιο σταθερού μήκους που ονομάζεται hash ενώ θα πρέπει:

- Να είναι εύκολη και γρήγορη στον υπολογισμό.
- Να είναι ντετερμινιστικές (Deterministic), δηλαδή η ίδια είσοδος θα πρέπει πάντα να επιστρέφει το ίδιο αποτέλεσμα.
- Να είναι υπολογιστικά αδύνατο να βρεθεί η αρχική είσοδος για ένα hash (Intractability).
- Να έχουν ισχυρή ανεκτικότητα σε συγκρούσεις (Collision-safety ή Collision resistance) δηλαδή να είναι υπολογιστικά αδύνατο να βρεθεί το ίδιο hash για δυο διαφορετικές εισόδους.
- Ακόμα και μικρές αλλαγές στην είσοδο να αλλάζουν σε τέτοιο βαθμό την έξοδο που δεν φαίνεται να υπάρχει καμία ομοιότητα μεταξύ των hashes.

Για παράδειγμα το αποτέλεσμα στο δεκαεξαδικό σύστημα αρίθμησης των συμβολοσειρών ‘Nikos Koutoslelos’ και ‘Nikos Koutsolelos ‘, δηλαδή ίδια συμβολοσειρά αλλά με κενό στο τέλος σε sha256 είναι:

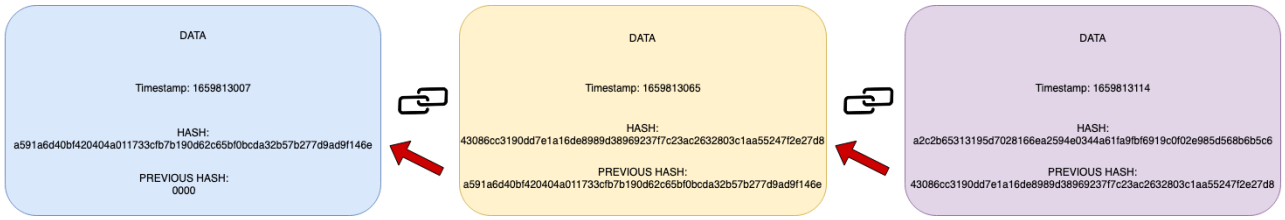
‘Nikos Koutsolelos’ : 32ccd3de22edb18071126d38384e6bae7f3cf2eb449bc69ecd346c953e9bbdce
‘Nikos Koutsolelos ‘: c7dd16630087db55b4e9770d62de29060720b6225a4d3aa99ed504cbb4ef3f64

Παρατηρούμε πως μια μικρή αλλαγή άλλαξε σημαντικά την έξοδο, επίσης ακόμα και με έναν περισσότερο χαρακτήρα στην είσοδο, το κενό στο τέλος, η έξοδος παραμένει σταθερή. Πρόκειται δηλαδή για συνάρτηση $h:\{0,1\}^* \rightarrow \{0,1\}^{\lambda}$, όπου μια είσοδος bit αυθαίρετου μήκους, μετατρέπεται σε έξοδος σταθερού μήκους λ.

Η συναρτήσεις κατακερματισμού είναι ένας τρόπος για να διαπιστώσουμε αν έχει τροποποιηθεί κάποια πληροφορία δημιουργώντας το λεγόμενο κρυπτογραφικό αποτύπωμα. Στο παραπάνω παράδειγμα, αν γνωρίζαμε πως το μήνυμα που έπρεπε να λάβουμε, είχε το πρώτο hash (32ccd…bdce) αλλά τρέχοντας τον sha256 είχαμε ως αποτέλεσμα το δεύτερο hash (c7dd…f64) τότε εύκολα θα διαπιστώναμε πως το μήνυμα έχει τροποποιηθεί. Για αυτό το λόγο το συναντάμε συχνά όταν κατεβάζουμε κάποιο αρχείο από το διαδίκτυο, ονομάζεται άθροισμα ελέγχου (checksum) και περιέχει το αποτέλεσμα κατακερματισμού του αρχείου, το οποίο μπορούμε να ελέγξουμε εκτελώντας την συνάρτηση κατακερματισμού τοπικά, χρησιμοποιώντας τον ίδιο αλγόριθμο, για να διαπιστώσουμε αν έχουμε το αυθεντικό ή όχι.

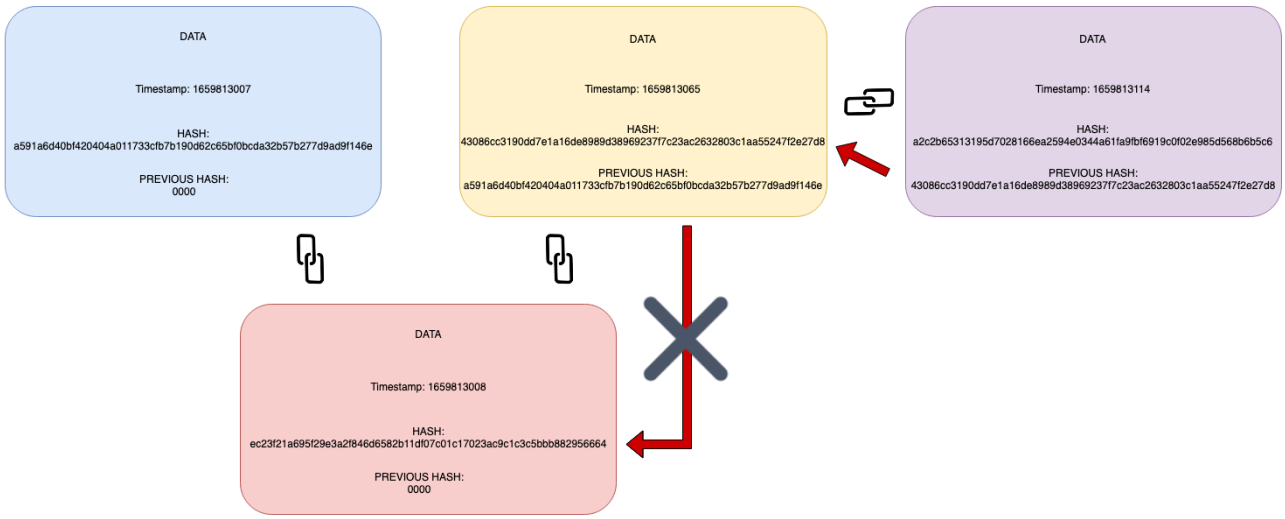
Ένα blockchain κάνει χρήση των hash functions για να δημιουργήσει ένα κρυπτογραφικό αποτύπωμα για κάθε block, αυτό το αποτύπωμα αποθηκεύεται μαζί με το block και καθορίζει τόσο τη μοναδική ταυτότητα του όσο και την ακεραιότητα των δεδομένων του. Το επόμενο block που δημιουργείται περιέχει το hash value του προηγουμένου στην αλυσίδα block, εκτός αν πρόκειται για το πρώτο στην αλυσίδα, το οποίο λέγεται και genesis block και περιέχει ένα μοναδικό hash (πχ. 0000).

Στο σχεδιάγραμμα που ακολουθεί βλέπουμε τα βασικά στοιχεία που πρέπει να περιέχει ένα block, τα δεδομένα, το timestamp, το hash όλων των δεδομένων που συγκρατεί και το hash του προηγούμενου block.



Σχήμα 8: Σύνθετο παράδειγμα με hash values ενός blockchain.

Για να αλλάξει το περιεχόμενο ενός block θα πρέπει δηλαδή να υπολογιστεί εκ νέου το hash του, καθώς αν δεν γίνει αυτό, το αποτέλεσμα του κατακερματισμού των δεδομένων του δεν θα συμφωνεί με το υπάρχον hash και στη συνέχεια να υπολογιστεί εκ νέου το hash κάθε block που το ακολουθεί. Το ίδιο θα πρέπει να γίνει αν θελήσουμε να προσθέσουμε καινούργιο block ενδιάμεσα.



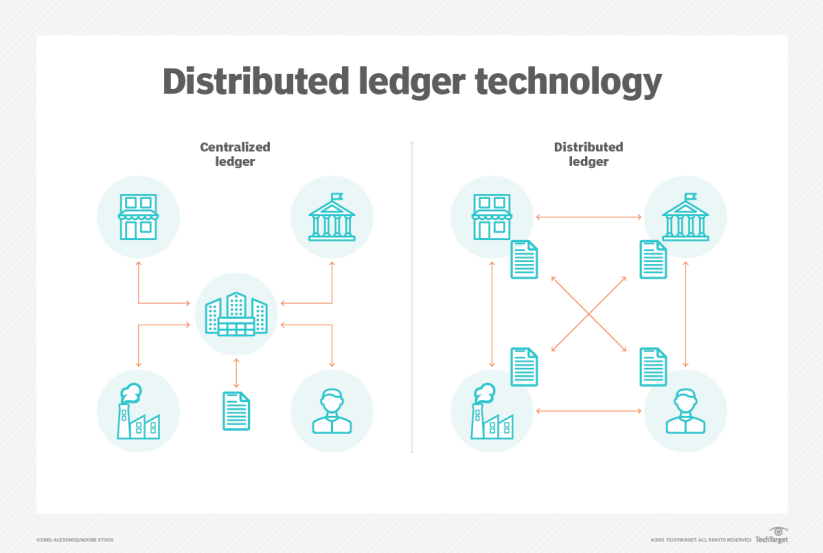
Σχήμα 9: Επίθεση σε ένα σύστημα blockchain προσθέτοντας ενδιάμεσο block.

Όπως φαίνεται στο παραπάνω σχεδιάγραμμα αν προσπαθήσουμε να προσθέσουμε ενδιάμεσο block (κόκκινο) αυτό δεν μπορεί να είναι μέλος αυτής της αλυσίδας, φαίνεται πως έχει γίνει τροποποίηση καθώς δεν ταιριάζουν οι τιμές των hash. Έτσι λοιπόν ένα σύστημα blockchain μπορεί μόνο να εμπλουτίζεται

συνεχώς με καινούργια blocks, δεν μπορεί να αλλάξει η σειρά, να τροποποιηθεί η υπάρχουσα πληροφορία, ή να διαγραφεί κάποιο από τα ήδη υπάρχοντα blocks. Εκτός δηλαδή από μια κατανεμημένη βάση δεδομένων, αποτελεί και ένα ακέραιο ιστορικό όλων των αλλαγών που υπήρξαν καθόλη τη ζωή της αλυσίδας.

2.3.3 Κατανεμημένο καθολικό

Μια βασική τεχνολογία που συχνά συνδυάζεται με το blockchain είναι αυτή του κατανεμημένου καθολικού (distributed ledger technology - DLT). Πρόκειται για μια βάση δεδομένων, τα δεδομένα της οποίας αντί να αποθηκεύονται σε μια κεντρική τοποθεσία, όπως ένας κλασικός διακομιστής (server), των οποίο διαχειρίζεται μια κεντρική αρχή, κατανέμονται σε ένα δίκτυο πολλών υπολογιστών που ονομάζουμε ομότιμο σύστημα κόμβων (peer-to-peer ή P2P) κάθε χρήστης ή node του οποίου διατηρεί αντίγραφο των δεδομένων. Όταν έρθει η ώρα να τροποποιηθούν τα δεδομένα, οποιοδήποτε node μπορεί να τροποποιήσει το δικό του καθολικό (αντίγραφο δεδομένων) και να ενημερώσει τα υπόλοιπα, στη συνέχεια με τη χρήση ενός αλγορίθμου συναίνεσης επιλέγεται ποιο αντίγραφο είναι το “σωστό”. Τέλος ενημερώνονται όλα τα nodes με αυτό το αντίγραφο, έτσι το δίκτυο είναι πάντα συγχρονισμένο.



Σχήμα 10: Διαφορά μεταξύ κατανεμημένης και κεντρικής βάσης δεδομένων
(Πηγή: www.techtarget.com)

Έτσι και σε ένα σύστημα blockchain δεν υπάρχει κεντρική αρχή η οποία διατηρεί τα δεδομένα, δεν υπάρχει πρωτότυπο (official copy), όλα τα nodes διατηρούν αντίγραφο των δεδομένων, είναι ισάξια και οποιοδήποτε από αυτά, ανάλογα με την υλοποίηση, μπορεί να προτείνει αλλαγή στα δεδομένα. Επιπρόσθετο πλεονέκτημα είναι ότι είναι δυσκολότερη η απώλεια δεδομένων, από λάθος, επίθεση ή φυσική καταστροφή, καθώς είναι σχεδιαστικά απαραίτητο αυτά να είναι αποθηκευμένα σε πολλαπλά nodes μέσα στο δίκτυο.

2.3.4 Δημόσια και ιδιωτικά blockchain

Μέχρι στιγμής αυτό που έχει αναλυθεί είναι οι αρχές ενός δημοσίου (public) blockchain καθώς αυτή ήταν η πρωταρχική ιδέα και οι μεγαλύτερες και πιο διαδεδομένες υλοποιήσεις της τεχνολογίας είναι δημόσιες αλυσίδες. Όπως αναφέρθηκε πρόκειται για μια πραγματικά αποκεντρωμένη εφαρμογή στην οποία δεν υπάρχει κεντρική αρχή, βασίζεται στην ύπαρξη πολλών node για την λειτουργία της και δεν ανήκει σε κανέναν. Μπορεί να συμμετέχει όποιος το επιθυμεί, όλοι οι συμμετέχοντες είναι ομότιμοι συνεπώς έχουν ίδια δικαιώματα. Υπάρχει διαφάνεια και μπορούν να δουν το blockchain, στο οποίο όταν γίνεται μια αλλαγή δεν μπορεί να αναστραφεί ή να διαγραφεί. Επίσης στην πλειοψηφία των περιπτώσεων είναι απαραίτητη η ύπαρξη και έκδοση κάποιου νομίσματος (token) προκειμένου να υπάρχει κάποια ανταμοιβή για την συμμετοχή και αφιέρωση πόρων στο δίκτυο. Τέλος αποκτά ασφάλεια λόγω του αλγορίθμου συναίνεσης που χρησιμοποιεί. Όμως αν και αυτή είναι η πιο διαδεδομένη υλοποίηση, δεν είναι η μοναδική.

Τα ιδιωτικά (private) blockchain εστιάζουν πιο πολύ στο απόρρητο της πληροφορίας, δεν είναι απαραίτητη η ύπαρξη κάποιου token για την λειτουργία τους, δεν μπορεί οποιοσδήποτε να συμμετέχει και ως εκ τούτου δεν μπορεί οποιοσδήποτε να δει ή να προσθέσει στη βάση, καθιστώντας απίθανη την επίθεση από εξωτερική απειλή. Μπορεί να επιτύχει μεγαλύτερη ταχύτητα στην επικύρωση καινούργιων block, τα οποία επικυρώνονται από εγγεγραμμένους χρήστες με κατάλληλα διαπιστευτήρια που είναι και μέλη του δικτύου. Επίσης λόγω του ότι χρησιμοποιούν άλλους αλγόριθμους συναινέσης, και έχουν συνήθως πολύ λιγότερους χρήστες, χρησιμοποιούν και πολύ λιγότερη ενέργεια. Υπονομεύουν ωστόσο κατά κάποιο τρόπο την λογική πίσω από ένα blockchain. Οι συμμετέχοντες μπορεί να μην έχουν ίσα δικαιώματα και συνήθως γνωρίζονται μεταξύ τους, συνεπώς δεν είναι τόσο αποκεντρωμένη υλοποίηση και ως εκ τούτου βασίζεται περισσότερο στην ακεραιότητα των χρηστών και του διαχειριστή [35].

2.3.5 Μηχανισμοί συναίνεσης

Ένα σύστημα blockchain χρειάζεται κάποιον μηχανισμό συναίνεσης. Αυτοί οι μηχανισμοί είναι υπαίτιοι για την ασφάλεια του συστήματος και την εγκυρότητα του. Επειδή το Blockchain δεν χρησιμοποιεί κάποια κεντρική αρχή για τη λειτουργία του, οι κατανεμημένοι ομότιμοι (peers) πρέπει να συμφωνήσουν για την εγκυρότητα των καινούργιων block. Υπάρχουν διάφοροι αλγόριθμοι συναίνεσης, ο καθένας εξυπηρετώντας τους σκοπούς του δικού του συστήματος, οπότε σε περίπτωση σύγκρισης όλοι έχουν τα πλεονεκτήματα και τα μειονεκτήματα τους προσπαθώντας να ισορροπήσουν μεταξύ ασφάλειας, λειτουργικότητας και επεκτασιμότητας. Εδώ θα γίνει αναφορά στους τρεις πιο γνωστούς αλγόριθμους για δημόσια και ιδιωτικά blockchain, το Proof of Work, το Proof of Stake και το Proof of Elapsed Time.

2.3.5.1 Proof of work (PoW)

Η ιδέα επινοήθηκε από τον Moni Naor και την Cynthia Dwork το 1993 [36] ως μια μέθοδος για την αποτροπή επιθέσεων τύπου Denial of service (DoS) ή την μαζική αποστολή ανεπιθύμητων μηνυμάτων (spam) καθώς απαιτεί κάποια εργασία (work) για την εκπλήρωση ενεργειών. Πρόκειται δηλαδή για έναν αλγόριθμο με τον οποίο ένα μέλος (αποδεικνύων ή prover) αποδεικνύει σε έναν άλλο (επιβεβαιωτής ή verifier) ότι έχει χρησιμοποιηθεί κάποια υπολογιστική ισχύς (work) για να εκπληρωθεί μια συγκεκριμένη ενέργεια. Ο επιβεβαιωτής με τη σειρά του μπορεί με ελάχιστη προσπάθεια, εύκολα και γρήγορα να επιβεβαιώσει αυτή τη δαπάνη πόρων. Αν υποθέσουμε πως ο αποστολέας ενός μηνύματος θα πρέπει να υπολογίζει και να επισυνάπτει στο μήνυμα του, το αποτέλεσμα μιας συνάρτησης κατακερματισμού του ίδιου του μηνύματος πριν την αποστολή του, η επίλυση της οποίας είναι δύσκολη και ότι ο παραλήπτης με τη σειρά του θα πρέπει να υπολογίσει την ίδια συνάρτηση για να διαπιστώσει το αμετάβλητο των δεδομένων που έλαβε, τότε και οι δυο πλευρές πρέπει να χρησιμοποιήσουν τους ίδιους πόρους, κάτι που μπορεί να είναι αποδεκτό για λίγα προσωπικά μηνύματα, αλλά όχι σε περιπτώσεις εταιριών ή διακομιστών αλληλογραφίας (email servers) που λαμβάνουν και πρέπει να επικαιροποιήσουν χιλιάδες μηνύματα. Συνεπώς ένα βασικό χαρακτηριστικό των αλγορίθμων τύπου proof of work είναι η ασυμμετρία στη χρήση υπολογιστικής ισχύος, η εργασία (work) πρέπει να είναι σχετικά δύσκολη ενώ ο έλεγχος πρέπει να είναι εύκολος. Ο σκοπός της εργασίας (work), δεν είναι η επίλυση του προβλήματος αυτού καθεαυτού αλλά η αποτροπή ανεπιθύμητων ή κακόβουλων ενεργειών, καθώς απαιτείται χρόνος, υπολογιστική ισχύς και κατα συνέπεια ηλεκτρική ενέργεια άρα και χρηματικό κόστος για την επιτυχή εκπλήρωση εργασιών ή ενεργειών.

Το 2002 ένας αλγόριθμος της ιδέας ονόματι Hashcash προτάθηκε από τον Adam Back [37] για το ίδιο πρόβλημα (DoS & Spam επιθέσεις). Ο αποστολέας έπρεπε να υπολογίσει και να αποστείλει στην κεφαλίδα του μηνύματος μια συμβολοσειρά [38] με την παρακάτω μορφή:

ver	:	bits	:	date	:	resource	:	ext	:	rand	:	counter
1	:	24	:	40806	:	foo	:		:	511801694b4cd6b0	:	1e7297a

Σχήμα 11: Κεφαλίδα αποστολέα

Όπου:

- ver: Ήταν η έκδοση του αλγορίθμου.
- bits: Ο ελάχιστος αριθμός των μηδενικών bit στο hash (pre-image).
- date: Η ημερομηνία που αποστέλλει το μήνυμα (timestamp).
- resource: Η ηλεκτρονική διεύθυνση (IP) ή το email του αποστολέα.
- ext: Η επέκταση αρχείου (extension) η χρήση της οποίας είναι προαιρετική για την έκδοση ένα.
- rand: Συμβολοσειρά τυχαίων χαρακτήρων, κωδικοποιημένη σε base-64.
- counter: Δυαδικός μετρητής, κωδικοποιημένος σε base-64.

Το Base-64 πρόκειται για κωδικοποίηση ψηφιακών δεδομένων σε κείμενο, αντιστοιχώντας τα byte ενός αρχείου σε υποσύνολο του ASCII.

Ο αποστολέας προετοιμάζει την κεφαλίδα (έκδοση, ώρα, IP, κτλ) και επισυνάπτει μια τυχαία τιμή (rand). Στη συνέχεια υπολογίζει το hash της κεφαλίδας αυτής και αν τα πρώτα 24 bit του αποτελέσματος είναι μηδέν, τότε βρέθηκε αποδεκτή κεφαλίδα και στέλνει το μήνυμα, διαφορετικά αυξάνει το μετρητή (counter), και προσπαθεί ξανά. Ο παραλήπτης με τη σειρά του υπολογίζει την συνάρτηση κατακερματισμού ολόκληρης της κεφαλίδας, ο υπολογισμός της οποίας είναι εύκολος και γρήγορος, απαιτώντας λιγότερο χρόνο και αποθηκευτικό χώρο από την λήψη του ίδιου του μηνύματος (message body). Με αυτό το τρόπο η δυσκολία εύρεσης αποδεκτής λύσης αυξάνεται εκθετικά με την αύξηση του αριθμού των επιθυμητών μηδενικών διπλασιάζοντας τον χρόνο, ενώ ο χρόνος επιβεβαίωσης παραμένει σταθερός.

Το 2009 μια παραμετροποιημένη υλοποίηση αυτού του αλγορίθμου proof of work χρησιμοποιήθηκε για την επικύρωση (validating ή minting) block στο bitcoin. Αντί για ένα προκαθορισμένο αριθμό μηδενικών bit στην αρχή του hash, αυτό ερμηνεύεται σαν ένας πολύ μεγάλος ακέραιος αριθμός, ο οποίος πρέπει να είναι μικρότερος από τον ακέραιο αριθμό που έθεσε σαν στόχο το δίκτυο (difficulty target). Αυτή η αλλαγή έγινε προκειμένου να είναι δυνατή η περιοδική προσαρμογή της δυσκολίας (περίπου κάθε 2,000 μπλοκ ή κάθε 14 μέρες), με σκοπό την διατήρηση ενός δεκάλεπτου, κατα μέσο όρο, διαστήματος ανά δημιουργία μπλοκ. Χωρίς αυτή την αλλαγή θα μπορούσε μόνο να διπλασιαστεί ή να μειωθεί κατα το ήμισυ η δυσκολία ακόμα και αν οι αλλαγές στο δίκτυο ήταν μικρές. Έτσι για κάθε καινούργιο υποψήφιο block τα μέλη ανταγωνίζονται για να βρουν έναν αριθμό (nonce) έτσι ώστε το hash των περιεχομένων του μπλοκ (nonce, data, previous block hash, κτλ) να ικανοποιούν το παραπάνω κριτήριο. Για να αλλάξει ένα block, πρέπει όχι μόνο να βρεθεί αποδεκτό nonce και να υπολογιστεί το hash του, αλλά να επαναληφθεί η ίδια διαδικασία για κάθε επόμενο block, κατά συνέπεια, οποιαδήποτε κακόβουλη αλλαγή μέσα στην αλυσίδα είναι υπερβολικά δύσκολη και χρονοβόρα. Ο Proof of Work έχει στιγματιστεί ως επιβλαβής για το περιβάλλον λόγω της ανταμοιβής και του ανταγωνισμού καθένας μέσα στο δίκτυο θέλει να είναι αυτός που επικυρώνει το μπλοκ, αυξάνοντας έτσι την υπολογιστική ισχύ που μπορεί να διαθέσει και κατά συνέπεια την ηλεκτρική ενέργεια που πρέπει να χρησιμοποιεί. Από έρευνα του πανεπιστημίου του Κέιμπριτζ [39] προκύπτει πως η συγκεκριμένη υλοποίηση χρειάζεται 88.8 τεραβατ/ώρα (TWh) το χρόνο ενώ σε σύγκριση με την ετήσια κατανάλωση χωρών προκύπτει [40]:

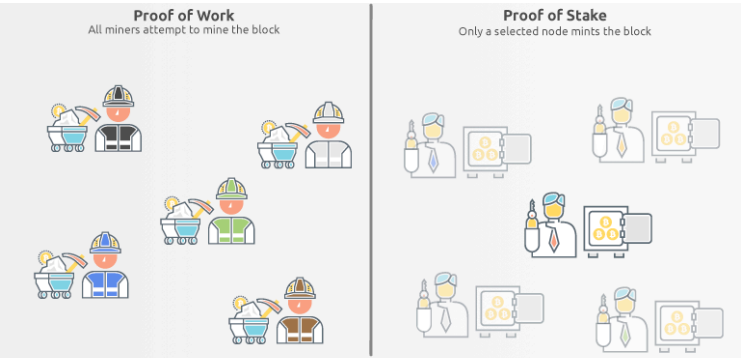
Χώρα	Κατανάλωση σε TWh
Bitcoin	88.8
Βέλγιο	83.5
Γερμανία	517.3
Ελλάδα	50.5
Ιαπωνία	903.7
Ηνωμένο Βασίλειο	304.1
Ηνωμένες Πολιτείες Αμερικής	3989.4

Σχήμα 12: Κατανάλωση ηλεκτρικής ενέργειας ανά χώρα για το 2019

Παρατηρούμε δηλαδή πως η κατανάλωση που χρειάζεται ο αλγόριθμος, στη κλίμακα που απαιτείται για ένα μεγάλο blockchain είναι σημαντική. Η μεγάλη κατανάλωση ενέργειας ωστόσο δεν αποτελεί μειονέκτημα της ίδιας της τεχνολογίας, αλλά του συγκεκριμένου αλγορίθμου συναίνεσης που λόγω της χρήσης του στη πρώτη και πιο γνωστή υλοποίηση στο κόσμο, ίσως επηρεάζει αρνητικά την κοινή γνώμη για την ίδια την τεχνολογία σε μια εποχή που ολοένα και αυξάνονται οι ανησυχίες για το περιβάλλον και τις άσκοπες χρήσεις ενέργειας.

2.3.5.2 Proof of Stake (PoS)

Το Proof of Stake είναι ένας αλγόριθμος που έχει σκοπό την μείωση της απαιτούμενης ενέργειας λειτουργίας του δικτύου. Σχεδιάστηκε για δημόσια blockchains που, όπως ειπώθηκε παραπάνω, συνήθως απαιτούν την ύπαρξη και έκδοση νομισμάτων (token) και κάνει χρήση ενός αλγορίθμου επιλογής αρχηγού (validator) που δεν απαιτεί από όλο το δίκτυο να ξοδέψει υπολογιστική ισχύ, για τον υπολογισμό κάποιου σύνθετου προβλήματος. Στην πιο απλή μορφή του, γίνεται μια κλήρωση ανάμεσα στα νομίσματα που έχουν δημιουργηθεί μέσα στο δίκτυο, ο κάτοχος του επιλεγμένου νομίσματος, μπορεί αν το επιθυμεί να προσθέσει στην αλυσίδα ένα μπλοκ. Οι πιθανότητες επιλογής ενός χρήστη, αυξάνονται ανάλογα με το μερίδιο που έχει, θεωρώντας ότι οι χρήστες με μεγάλο μερίδιο του πλούτου μέσα στο δίκτυο, έχουν μεγαλύτερο κίνητρο για την ορθή λειτουργία του [41]. Το πρόβλημα με αυτή την λύση είναι ότι δεν υπάρχει ποινή για κακόβουλες ενέργειες, δεν έγινε δαπάνη πόρων (ηλεκτρική ενέργεια, υπολογιστική ισχύς, χρόνος, κτλ) για τον υπολογισμό κάποιου σύνθετου hash, συνεπώς δεν υπάρχει κάποιος αποτρεπτικός παράγοντας για την διεξαγωγή κακόβουλων ενεργειών. Η λύση για αυτό το πρόβλημα είναι να υπάρχει ένα ποσό που δεσμεύεται (stake) από τον ή τους υποψήφιους validators. Αν βρεθεί πως επιχείρησε να προσθέσει κακόβουλο μπλοκ μέσα στην αλυσίδα, χάνει το stake (Slashing) και δεν λαμβάνει την ανταμοιβή. Ο σκοπός δηλαδή είναι να αντικατασταθεί η δαπάνη υπολογιστικής ισχύος και χρόνου, με την απώλεια χρημάτων [42]. Σε σύγκριση με το Proof of Work, το Proof of Stake επιβαρύνει λιγότερο το περιβάλλον δεν απαιτεί τη συνεχή χρήση υπολογιστικής δύναμης από όλο το δίκτυο, αλλά μόνο αυτού που επιλέγεται για την επικύρωση του καινούργιου block.



Σχήμα 13: PoW όπου όλοι οι συμμετέχοντες προσπαθούν να επικαιροποιήσουν το επόμενο μπλόκ και PoS όπου μόνο ο/οι επιλεγμένος/οι από το δίκτυο επικαιροποιεί το επόμενο μπλοκ (Πηγή: nicehash.com)

2.3.5.3 Proof of Elapsed Time (PoET)

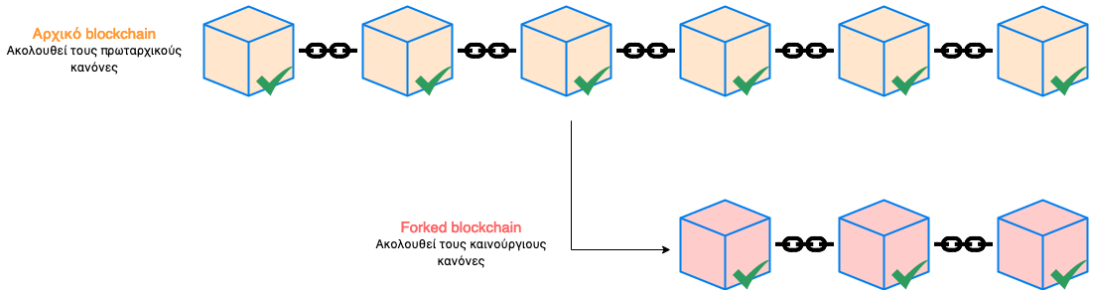
Το 2015 το Linux Foundation ξεκίνησε το Hyperledger ένα πρότζεκτ με σκοπό την ανάπτυξη εργαλείων blockchain προς όφελος όλων. Το 2016 συνέβαλε η Intel, μια από της μεγαλύτερες εταιρίες σχεδίασης και κατασκευής επεξεργαστών, δημιουργώντας το Hyperledger Sawtooth, μια αλυσίδα που είχε τη δυνατότητα να αλλάζει αλγόριθμο συναίνεσης ανάλογα με τις απαιτήσεις του δικτύου. Ανάμεσα στις επιλογές που είχε, σχεδιάστηκε ένας καινούργιος αλγόριθμος, ο Proof of Elapsed Time. Σχεδιασμένος για ιδιωτικές αλυσίδες που έχουν πολύ μεγάλη κίνηση πληροφοριών και χρειάζονται ένα σύστημα που εγγυάται το αμετάβλητο των δεδομένων τους. Παράλληλα μπορεί να υλοποιηθεί σε συστήματα διατηρώντας την επεκτασιμότητα (scalability) και ελαχιστοποιώντας τους απαιτούμενους πόρους και την κατανάλωση ηλεκτρικής ενέργειας. Κάνει χρήση ενός συστήματος δίκαιης λοταρίας χρησιμοποιώντας την γεννήτρια τυχαίων αριθμών ή Hardware random number generator (HRNG) ή true random number generator (TRNG) η οποία, σε αντίθεση με τους αλγόριθμους που χρησιμοποιούνται συνήθως για την παραγωγή τυχαίων αριθμών (pseudo-random number) είναι ικανή να παράγει όσο το δυνατόν πιο τυχαίους αριθμούς με τη χρήση τυχαίων φαινομένων εντός του επεξεργαστή. Η διαδικασία αυτή τρέχει σε ένα ασφαλές περιβάλλον ή Trusted Execution Environment (TEE) προκειμένου κάθε node να δημιουργήσει ένα τυχαίο χρόνο για τον οποίο θα περιμένει ανενεργό (sleep time). Όταν τελειώσει αυτός ο χρόνος, ξεκινάει η διαδικασία επικαιροποίησης του καινούργιου block το οποίο και αποστέλλεται σε όλο το δίκτυο όταν ολοκληρωθεί.

Σχήμα 14: Διαδικασία δημιουργίας Block με χρήση PoET (Πηγη: geeksforgeeks)

Το σύστημα βασίζεται στις τεχνολογίες που αναπτύχθηκαν, από την intel για τους επεξεργαστές της επιτρέποντας την δαπάνη πόρων μόνο από τα λίγα αυτά node μέσα στο δίκτυο που ξύπνησαν πρώτα . Η ίδια λογική ωστόσο μπορεί να τροποποιηθεί και να αναπτυχθεί και σε άλλα συστήματα.

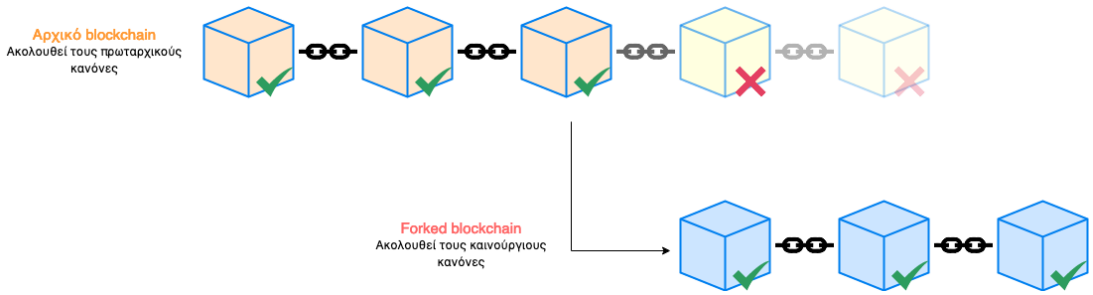
2.3.6 Διακλαδώσεις στα blockchain

Σε ένα blockchain μπορεί να υπάρξουν δυο τύποι διακλαδώσεων (fork), τα λεγόμενα hard forks και τα soft forks. Ένα fork μπορεί να προκύψει λόγω αναβαθμίσεων, όπως για παράδειγμα αλλαγή/βελτίωση των κανόνων, αναβάθμιση του κώδικα με σκοπό την προσθήκη νέων λειτουργιών, ενημέρωση του πρωτοκόλλου ασφαλείας ή λόγω διαφωνίας εντός του δικτύου. Στην περίπτωση ενός hard fork οι αλλαγές είναι σημαντικές έχοντας ως αποτέλεσμα πολλές φορές την δημιουργία δυο ξεχωριστών blockchain, μια με τους καινούργιους κανόνες και μια με τους πρωταρχικούς κανόνες που συνεχίζει κανονικά, σαν να μην υπήρξε ποτέ αναβάθμιση.



Σχήμα 14: Παράδειγμα ενός hard forked blockchain

Στην περίπτωση ενός soft fork οι αλλαγές δεν είναι τόσο ριζοσπαστικές, πρόκειται για αναβαθμίσεις οι οποίες είναι συμβατές με την τρέχουσα αλυσίδα και τα μπλοκ της και έτσι συνήθως δεν χωρίζεται το blockchain σε δυο ξεχωριστές αλυσίδες. Επιπροσθέτως ένα soft fork μπορεί να προκύψει αν δυο nodes επικυρώσουν ταυτόχρονα ένα καινούργιο μπλοκ, το οποίο μπορεί να συμβεί και λόγω καθυστερήσεων στο δίκτυο [43] ή αν κάποιο node προσπαθήσει να προσθέσει κάποιο κακόβουλο μπλοκ.



Σχήμα 15: Παράδειγμα ενός soft forked blockchain

Στο παραπάνω σχήμα μετά το τελευταίο πορτοκαλί μπλοκ, δημιουργείται προσωρινό fork με δυο διακλαδώσεις. Αν κριθεί ότι και τα δυο μπλοκ είναι έγκυρα (κίτρινο και μπλε μπλοκ), τότε ποια διακλάδωση θα επικρατήσει εξαρτάται από την υλοποίηση, για παράδειγμα σε ένα PoW δίκτυο μπορεί να επιλέγεται το fork στο οποίο έχουν δαπανηθεί περισσότεροι πόροι όπως φαίνεται και στο σχήμα. Αν πρόκειται για αναβάθμιση, τότε καθώς ολοένα και περισσότερα node επικυρώνουν μπλοκ βάσει των καινούργιων κανόνων, αυτό θα είναι και το fork που θα επικρατήσει.

2.3.7 Χαρακτηριστικά μίας εφαρμογής Blockchain

Συνοψίζοντας, οι ιδιότητες που προσφέρονται από μια βασική εφαρμογή blockchain κατανεμημένου καθολικού είναι:

- Αποκέντρωση: Το βασικό γνώρισμα της τεχνολογίας blockchain είναι ότι η προσθήκη σε αυτό δεν απαιτεί την χρήση κάποιας τρίτης επαλήθευσης. Τα δεδομένα αποθηκεύονται σε ένα καθολικό (ledger) το οποίο διαμοιράζεται σε όλους τους χρήστες/ομότιμους (peers ή nodes) του δικτύου. Αυτό το καθολικό (ledger) αποθηκεύει όλο το ιστορικό.
- Αντίγραφο δεδομένων: Όλοι οι χρήστες του blockchain έχουν το ίδιο αντίγραφο καθολικού, έτσι ανά πάσα στιγμή υπάρχουν πολλαπλά αντίγραφα της βάσης.
- Διαφάνεια Δεδομένων: Κάθε χρήστης του δικτύου, μπορεί να δει τα μπλοκ, και να συμμετέχει στην δημιουργία νέων blocks στο πέρασμα του χρόνου.
- Αμετάβλητο δεδομένων: Λόγω της διαδικασίας του κατακερματισμού, τη σύνδεση μεταξύ των nodes και του διαμοιρασμού του καθολικού σε όλους τους χρήστες, διασφαλίζεται το αμετάβλητο των δεδομένων [44].

Τα παραπάνω ωστόσο, μπορούν να τροποποιηθούν ανάλογα με την υλοποίηση (private ή public blockchain).

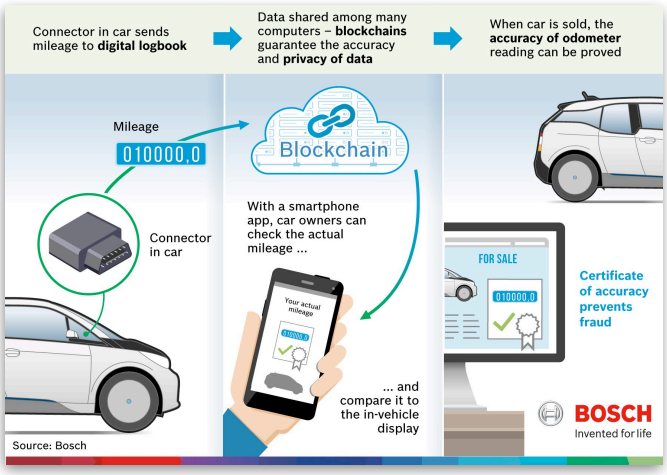
Αν δηλαδή έχουμε ένα κατανεμημένο σύστημα με node που διατηρούν ένα κατανεμημένο καθολικό, στο οποίο μπορούν μόνο να προστεθούν δεδομένα τα οποία συνδέονται μεταξύ τους και έτσι διατηρείται ένα απροσπέλαστο ιστορικό, τότε μπορούμε να πούμε ότι έχουμε ένα σύστημα blockchain.

2.4 Εφαρμογές του Blockchain

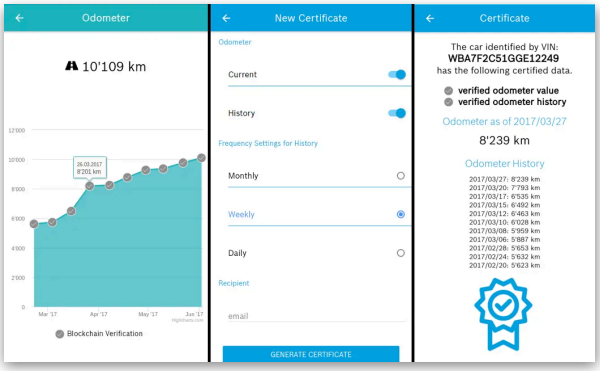
Τα πλεονεκτήματα που προσφέρει η τεχνολογία Blockchain, κυρίως όσον αφορά την διασφάλιση των δεδομένων, έχουν αρχίσει να παρατηρούνται και να χρησιμοποιούνται μέσω της υιοθέτησης της σε διάφορες εφαρμογές. Η χρήση αποκεντρωμένης τεχνολογίας, μπορεί να ενισχύσει τη διαφύλαξη των πληροφοριών, παρεμποδίζοντας ή μειώνοντας την πιθανότητα επιτυχημένων επιθέσεων από εισβολείς [45] που επιδιώκουν να τροποποιήσουν δεδομένα.

2.4.1 Παραποίηση του οδομέτρου αυτοκινήτου Bosch IoT Lab (odometer fraud)

Η μεταπωλητική αξία ενός αυτοκινήτου εξαρτάται, μεταξύ άλλων, και από τα χιλιόμετρα που έχει διανύσει για αυτό όλο και πιο σύνηθες είναι το φαινόμενο του γυρίσματος των χιλιομέτρων. Υπολογίζεται πως το 30% με 50% των αυτοκινήτων που πωλούνται εντός της Ε.Ε έχουν χειραγωγημένα οδόμετρα [46]. Σε αυτό το πρόβλημα έχουν προτείνει λύσεις κάποια κράτη μέλη, όπως το Βέλγιο με την υπηρεσία Car-Pass [47]. Το συγκεκριμένο σύστημα βασίζεται στην ενημέρωση μιας βάσης με τα πραγματικά χιλιόμετρα του αυτοκινήτου, από αξιόπιστες πηγές όπως ιδιοκτητών χώρων στάθμευσης, συνεργεία αυτοκινήτων, τεχνικός έλεγχος (ΚΤΕΟ), κτλ. Με αυτό το τρόπο η βάση ενημερώνεται πιο συχνά με αξιόπιστα δεδομένα, καθιστώντας δυσκολότερη την απάτη. Ακόμα όμως και αυτό το σύστημα έχει προβλήματα, καθώς δεν μπορεί πάντα να εγγυηθεί η αξιοπιστία αυτών των επιχειρήσεων / θεσμών / ατόμων, είτε από δωροδοκίες είτε από αμέλεια, αδιαφορία, απροσεξία ή λάθους. Η Bosch σε συνεργασία με τα πανεπιστήμια του Σεν Γκάλεν και της Ζυρίχης της Ελβετίας βλέποντας τις δυνατότητες των καινούργιων τεχνολογιών υλοποίησε ένα σύστημα που βασίζεται στο blockchain και στο διαδίκτυο των αντικειμένων (Internet of things - IoT) [48]. Το ίδιο το αυτοκίνητο θα στέλνει τα πραγματικά χιλιόμετρα που έχει διανύσει σε ένα blockchain ανά τακτά χρονικά διαστήματα, εγγυώντας τη γνησιότητα των στοιχείων, και κάνοντας εύκολο τον έλεγχο των πραγματικών χιλιομέτρων του αυτοκινήτου από τον ενδεχόμενο αγοραστή.



Σχήμα 16: Σχεδιάγραμμα λειτουργίας του συστήματος καταγραφής χιλιομέτρων
(Πηγή: Bosch Iot lab)



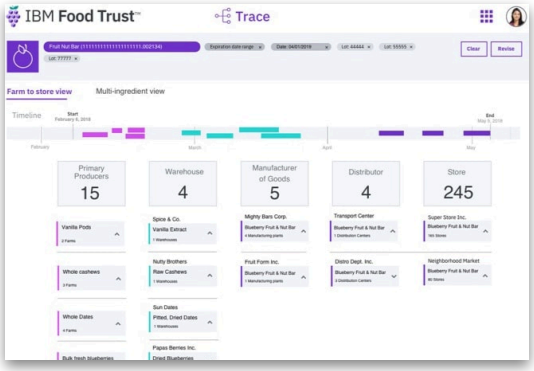
Σχήμα 17: Η εφαρμογή κινητού, μέσα από την οποία ο χρήστης μπορεί να αναζητήσει το ιστορικό των μετρήσεων.
(Πηγή: Bosch Iot lab)

Το μόνο που θα πρέπει να κάνει ο αγοραστής είναι να κατεβάσει την εφαρμογή στο κινητό, και να αναζητήσει το VIN (Vehicle identification number) του αυτοκινήτου. Εκτός από τα συνολικά χιλιόμετρα, είναι διαθέσιμο και το ιστορικό των μετρήσεων (χιλιόμετρα και ημερομηνία λήψεις μέτρησης) κάνοντας ακόμα πιο δύσκολη την αλλαγή τους.

2.4.2 Το Blockchain στον τομέα του Supply Chain και Logistics - IBM

Σήμερα μπαίνοντας σε ένα κατάστημα λιανικής πώλησης (σούπερ μάρκετ) έχουμε την δυνατότητα να προμηθευτούμε μια τεράστια γκάμα προϊόντων και τροφίμων για τα οποία υπάρχουν νόμοι και εργαστηριακές εξετάσεις που διασφαλίζουν την ποιότητα τους, ωστόσο μερικές φορές, υπάρχει η ανάγκη να αποσυρθεί ένα προϊόν και να ενημερωθεί το κοινό. Τότε είναι σημαντικό να βρεθεί η προέλευση του όσο πιο γρήγορα γίνεται προκειμένου να εξασφαλιστεί η δημόσια υγεία και να μειωθούν οι σπατάλες και το γενικότερο κόστος. Ανάλογα με το προϊόν ωστόσο, μπορεί να χρειάζεται η συνεργασία πολλών για να εξακριβωθεί η προέλευση του (φάρμα, μεταφορική εταιρία, εργοστάσιο, συσκευαστές, κ.α) απαιτώντας πολύτιμο χρόνο. Το 2016 η εταιρία Wallmart προσπάθησε να βρει την προέλευση μιας συσκευασίας κομμένων φρούτων, χρειάστηκαν 6 μέρες, 18 ώρες, και 26 λεπτά [49]. Όταν βρεθεί πρόβλημα με ένα προϊόν, οι αρχές ζητούν απόσυρση ή αποφυγή αγοράς όλων των προϊόντων μιας περιοχής ή ακόμα και άλλων προϊόντων της ίδιας κατηγορίας, όχι μόνο αυτών που προέρχονται από την επηρεασμένη φάρμα ή εργοστάσιο [50]. Μια άλλη λύση είναι η απόσυρση μιας ολόκληρης παρτίδας με τη χρήση κάποιου κωδικού, ωστόσο και αυτό δεν είναι ιδανικό καθώς μπορεί να μην έχει επηρεαστεί ολόκληρη η παρτίδα ή μπορεί να μην είναι η μόνη επηρεασμένη.

Έτσι λοιπόν η IBM ανέπτυξε το Food Trust [51] ένα σύστημα blockchain βασισμένο στο Hyperledger Fabric (Linux Foundation) που επιτρέπει την καταχώρηση πληροφοριών για οπωροκηπευτικά, κρεατικά, γαλακτοκομικά και πολυσυστατικά προϊόντα. Λόγω της φύσης του blockchain, κάθε στάδιο μπορεί να εμπλουτίσει την πλατφόρμα με πληροφορίες για τα προϊόντα που καλλιέργησε, έλαβε, τροποποίησε ή συσκεύασε. Έτσι είναι δυνατή η παρακολούθηση τους, διατηρώντας πληροφορίες για όλα τα προϊόντα και τα συστατικά τους (τοποθεσία προέλευσης, ημερομηνία, που βρίσκονται, πως χρησιμοποιήθηκαν, που πουλήθηκαν, κ.α).

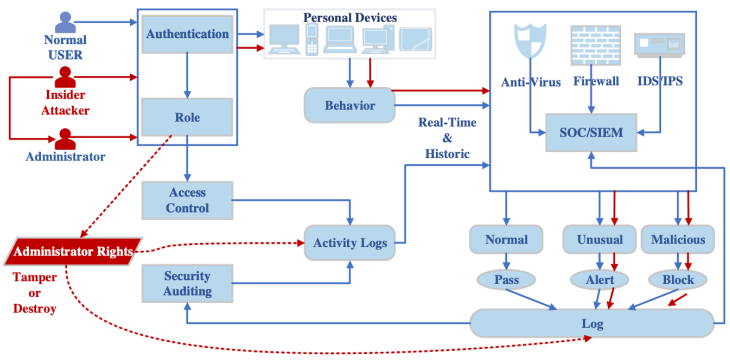


Σχήμα 18: Σύστημα blockchain της IBM (Πηγή [51])

Στην περίπτωση που βρεθεί ότι ένα μη συσκευασμένο προϊόν ή ένα συστατικό ενός συσκευασμένου προϊόντος είναι προβληματικό, είναι εύκολη και γρήγορη (22 δευτερόλεπτα σε σχέση με 6 μέρες [50]) η εύρεση της προέλευσης του, και να αποσυρθούν όλα τα επηρεασμένα και μόνο προϊόντα.

Κεφάλαιο 3. Συστήματα Ανίχνευσης Εισβολών Με Τεχνολογία Blockchain

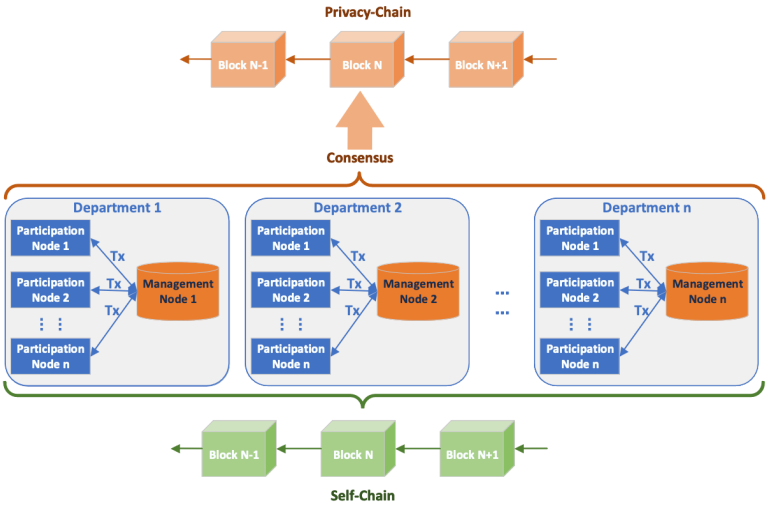
Γίνεται αντιληπτό, έχοντας αναλύσει την τεχνολογία του blockchain πως αυτή μπορεί να χρησιμοποιηθεί σαν πολύτιμο επιπρόσθετο εργαλείο σε συνεργασία με υπάρχουσες τεχνολογίες όπως αυτής των συστημάτων ανίχνευσης εισβολών προκειμένου να δημιουργηθεί ένα ολοκληρωμένο σύστημα ασφαλείας. Οπως αναλύσαμε και σε προηγούμενη ενότητα, αν και τα συστήματα ανίχνευσης εισβολών αποτελούν ένα απραίτητο εργαλείο, ειδικότερα στην περίπτωση εσωτερικής επίθεσης, είναι σημαντικό να λάβουμε υπόψιν πως ο επιτιθέμενος όχι μόνο έχει πρόσβαση μέσα στο σύστημα με αυξημένα δικαιώματα, αλλά μπορεί και να γνωρίζει όλες τις πτυχές του συστήματος (τοπολογία δικτύου, αδυναμίες συστήματος ή/και admin κωδικούς πρόσβασης). Έτσι γίνεται πιο εύκολη τόσο η επίθεση όσο και η αποφυγή ενεργοποίησης συναργεμών από τα συστήματα ασφαλείας, διαγράφοντας ή τροποποιώντας τα δεδομένα που λαμβάνουν υπόψιν τα σύστηματα αυτά (junk in junk out) ή ακόμα και απενεργοποιώντας τα. Μετά το πέρας της επίθεσης, πιθανό είναι το σενάριο της διαγραφής μέρους ή ολοκλήρου του ιστορικού προκειμένου να καταστραφούν τα ενοχοποιητικά στοιχεία (διαγραφή logs). Σε αυτό το κομμάτι δύναται να βοηθήσει ένα σύστημα blockchain με κατανεμημένο καθολικό. Συγκεκριμένα η κρυπτογραφική αλυσίδα της τεχνολογίας διασφαλίζει ότι από την στιγμή που καταγράφονται δεδομένα είναι πρακτικά αδύνατο να τροποποιήθουν, παρέχοντας ένα ασφαλές αρχείο δεδομένων και ψηφιακών αποδεικτικών επιτρέποντας την ικανότητα διατήρησης αμετάβλητων στοιχείων [52]. Επιπροσθέτως η κατανεμημένη φύση του μπορεί να διατηρήσει δεδομένα σε node πέρα από το δίκτυο της εταιρείας, πιθανώς σε άλλα υποκαταστήματα κάνοντας ακόμα πιο δύσκολη τη διαγραφή των δεδομένων που διατηρεί. Τα δεδομένα αυτά μπορούν σε πρώτο χρόνο να χρησιμοποιηθούν από το IDS για την αναγνώριση και αναχαίτιση επιθέσεων και σε δεύτερο χρόνο, εφόσον καταστεί αναγκαίο, την ανάλυση της επίθεσης. Ο συνδυασμός ωστόσο των δυο αυτών τεχνολογιών δεν αποτελεί καινούργια ή πρωτότυπη ιδέα. Το 2020 ερευνητές του University of Electronic Science and Technology [53] της Κίνας, αναγνωρίζοντας τα πιθανά πλεονεκτήματα που μπορεί να προσφέρει η τεχνολογία σχεδίασαν ένα σύστημα με σκοπό την διατήρηση των δεδομένων ιχνηλατήσεις (logs) σε περίπτωση επίθεσης από χρήστη με απόλυτα δικαιώματα στο σύστημα. Έχοντας τα δικαιώματα αυτά μετά το πέρας της επίθεσης και χωρίς να έχει αυτή γίνει αντιληπτή από τα συστήματα ανίχνευση ή την ομάδα ασφαλείας, ο χρήστης αυτός θα μπορούσε να διαγράψει πολύτιμες πληροφορίες για την επίθεση που πραγματοποίησε κάνοντας την ανίχνευση ή την ανάλυση της επίθεσης πολύ πιο δύσκολη ή και αδύνατη.



Σχήμα 19: Σχεδιάγραμμα ανάλυσης δικτύου και επιθέσεις από εσωτερικά επιτιθέμενο χρήστη.

Στο παραπάνω σχεδιάγραμμα φαίνεται η τοπολογία του δικτύου, ο χρήστης εκτελεί κανονικά νόμιμες ενέργειες (μπλε γραμμές) και κακόβουλες ενέργειες (κόκκινες γραμμές) που όμως είναι επιτρεπτές από το σύστημα, εκτελούνται κανονικά και δεν γίνονται αντιληπτές από τα συστήματα ασφαλείας. Στη συνέχεια μπορεί να διαγράψει τα logs. Η λύση που πρότειναν ήταν ένα σύστημα blockchain που εφαρμόζεται στο δίκτυο μίας εταιρίας και αποτελείται από δυο τύπους κόμβων, τους διαχειριστικούς (management nodes) και τους απλούς (participation nodes) καθένας από τους οποίους διατηρεί δυο αλυσίδες, μια λεγόμενη privacy-chain και μια self-chain. Κάθε τμήμα έχει διαφορετικά δικαιώματα, ως εκ τούτου οι χρήστες κάθε τμήματος θα πρέπει να διαχειρίζονται από το τμήμα στο οποίο ανήκουν.

Ο διαχειριστικός κόμβος είναι ο κύριος κόμβος κάθε τμήματος, είναι υπεύθυνος για την δημιουργία καινούργιων block, την επαλήθευση των δεδομένων που λαμβάνει από τους απλούς κόμβους (logs) και την διατήρηση των privacy και self chain του τμήματος. Οι απλοί κόμβοι διατηρούν και αυτοί με τη σειρά τους δυο αλυσίδες (privacy και self). Πρόκειται για χρήστες του συστήματος, που καταγράφουν και αποστέλλουν τα logs τους στους διαχειριστικούς κόμβους του τμήματος που ανήκουν. Έτσι κάθε τμήμα έχει ένα διαχειριστικό κόμβο και πολλούς απλούς κόμβους. Η αλυσίδα self-chain που διατηρεί κάθε κόμβος περιέχει όλα τα στοιχεία που αφορούν τις ενέργειες που έκανε ο χρήστης, ενώ η self-chain του κόμβου διαχειρίσεις, περιέχει τις ενέργειες όλων των χρηστών του τμήματος της. Η κυρία αλυσίδα που διατηρεί κάθε κόμβος είναι η κυρία αλυσίδα του συστήματος, όπου όλοι οι κόμβοι συμμετέχουν και διατηρούνται δεδομένα όπως τμήμα, χρήστης, χρονοσήμανση και πληροφορίες σχετικές με το ιστορικό των ενεργειών που έκαναν οι χρήστες οι οποίες, με την χρήση, παραμένουν κρυφές.

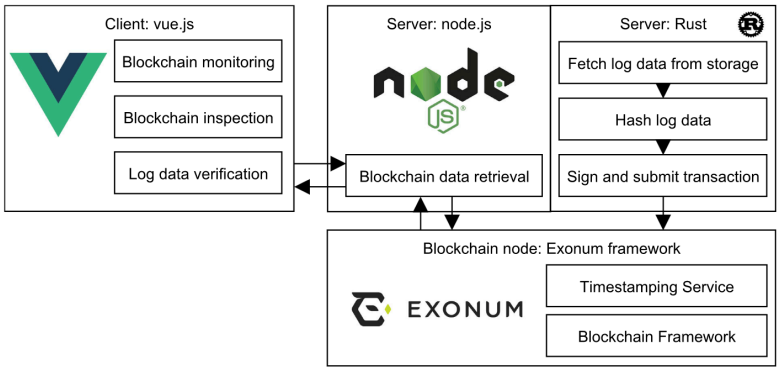


Σχήμα 20: Σχηματική απεικόνιση κόμβων και αλυσίδας συστήματος

Συνεπώς υπάρχει μόνο μια κύρια αλυσίδα privacy-chain στην οποία συμμετέχουν όλοι οι κόμβοι και υπάρχουν πολλές δευτερεύουσες αλυσίδες (self-chain) καθώς κάθε κόμβος διατηρεί μια δικιά του. Με την χρήση δυο αλυσίδων το σύστημα είναι ικανό να καταγράψει τα απαραίτητα δεδομένα, διασφαλίζοντας την ανωνυμία. Ακόμα και μέσα σε ένα ιδιωτικό δίκτυο δεν υπάρχει λόγος, καθώς η αλυσίδα είναι προσβάσιμη από όλους, να μπορούν να δουν όλα τα τμήματα τις ενέργειες όλων των άλλων. Ο αλγόριθμος συναίνεσης που ανάπτυξαν ήταν τροποποιημένος DPoS (Delegated Proof of Stake) αλλά αντί να χρειάζεται να εκλέγεται ένας κόμβος, χρησιμοποιούνται οι διαχειριστικοί κόμβοι οι οποίοι επιλέγονται τυχαία για την επικύρωση του

καινούργιου block. Έτσι δεν χρειάζεται η έκδοση νομίσματος και μειώνονται οι απαιτήσεις πόρων και χρόνου. Τα ευρήματα τους ήταν θετικά καθώς το σύστημα ήταν σε θέση να διαφυλάξει τα δεδομένα. Ωστόσο επισήμαναν την ανάγκη για περαιτέρω διερεύνηση καθώς η εφαρμογή ενός τέτοιου συστήματος σε ένα μεγάλο δίκτυο μπορεί να επιφέρει μεγάλο φόρτο στα συστήματα και κατά συνέπεια καθυστερήσεις. Επίσης σημείωσαν την περίπτωση όπου το δίκτυο είναι πολύ μικρό με λίγους κακόβουλους κόμβους, οι οποίοι σε συνεργασία θα μπορούσαν να τροποποιούν τα δεδομένα που προστίθενται στην αλυσίδα.

Η παραπάνω υλοποίηση δεν αποτελεί την μόνη προσπάθεια ένταξης του blockchain σε ένα υπολογιστικό σύστημα με σκοπό την αύξηση της ασφάλειας του. Το 2019 ερευνητές από το Πανεπιστήμιο του Ρέγκενσμπουργκ [54] αναγνωρίζοντας την ανάγκη διατήρησης αποδεικτικών στοιχείων μετά το πέρας μιας επίθεσης για χρήση στο δικαστήριο, ανέπτυξαν ένα σύστημα που βασιζόταν σε μια υπάρχουσα blockchain την EXONUM.



Σχήμα 21: Σχεδίαση Client-Server εφαρμογής

Τα συμπεράσματα και αυτών ήταν θετικά καθώς το σύστημα ήταν σε θέση να καταγράψει και να διασφαλίσει την εμπιστευτικότητα των δεδομένων με τρόπο τέτοιο ώστε αυτά να μπορούν να κατατεθούν σε δικαστήριο ως αποδεικτικά στοιχεία, ωστόσο έθεσαν το ερώτημα της ασφαλείας της ίδιας της αλυσίδας, ειδικά λόγω του ότι η υλοποίηση τους βασιζόταν σε υπάρχον blockchain το οποίο δεν είχε επικυρωθεί για την ασφάλεια του από κανέναν. Πέρα από αυτό αναφέραν την πολυπλοκότητα προστίθεται στο σύστημα, αλλά και πως μπορούν να υπάρξουν προβλήματα με την επεκτασιμότητα του συστήματος. Σε ένα πραγματικό σενάριο, με χιλιάδες χρήστες και δεδομένα θα ήταν πρακτικά αδύνατο να διατηρηθεί μια αλυσίδα στην οποία συνεχώς προστίθενται δεδομένα, αργά ή γρήγορα θα υπήρχαν προβλήματα με το χώρο.

Αναγνωρίζοντας τα πιθανά πλεονεκτήματα που μπορεί να προσφέρει η τεχνολογία blockchain όταν συνδυαστεί με ένα σύστημα ασφαλείας σχεδιάστηκε και υλοποιήθηκε πρακτική εφαρμογή προκειμένου την καλύτερη κατανόηση της τεχνολογίας και την διερεύνηση της βιωσιμότητας του συνδυασμού των δυο αυτών τεχνολογιών. Στο κεφάλαιο που ακολουθεί θα γίνει αναλυτική περιγραφή του σεναρίου, της εφαρμογής και των τεχνολογιών που χρησιμοποιήθηκαν. Εν συνέχεια θα γίνουν ενδεικτικές επιθέσεις στο σύστημα προκειμένου να κριθεί η αποτελεσματικότητα του.

Κεφάλαιο 4. Σχεδιασμός Και Υλοποίηση Συστήματος

4.1 Ο κόσμος του προβλήματος

Προκειμένου να εξεταστεί η αποτελεσματικότητα της τεχνολογίας IDS με blockchain θεωρήθηκε το σενάριο μιας βάσης δεδομένων, με πολλούς χρήστες καθένας έχοντας διαφορετικά δικαιώματα, όπως θα γινόταν στην περίπτωση κάποιας πολυεθνικής εταιρείας. Έτσι είναι δυνατόν να δοκιμάσουμε επιθέσεις με διαφορετικά επίπεδα τεχνικής κατάρτισης και πρόσβασης στη βάση βλέποντας την αποτελεσματικότητα του συστήματος. Η εν λόγω βάση περιέχει ονοματεπώνυμα, διευθύνσεις ηλεκτρονικής αλληλογραφίας, νούμερα τηλεφώνων, ημερομηνία πρόσληψης, τμήμα, περιγραφή εργασίας, μισθολογικά δεδομένα και ημερομηνία αλλαγής μισθού για το πλήθος των εργαζόμενων που απαρτίζουν την εταιρία. Η στήλη ημερομηνίας αλλαγής μισθού αποθηκεύει, όπως άλλωστε υποδεικνύει και το όνομα της, την ημερομηνία που άλλαξε ο μισθός.

FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	DEPARTMENT	JOB	SALARY	SALARY_DATE
John	Doe	john.doe@example.com	5551234567	2023-01-15	Engineering	Software	60000.00	2023-08-19
Jane	Smith	jane.smith@example.com	5559876543	2022-11-20	Sales	Manager	80000.00	2023-08-19
Michael	Johnson	michael.johnson@example.com	5557890123	2023-03-10	Marketing	Specialist	55000.00	2023-08-19
Sarah	Williams	sarah.williams@example.com	5552345678	2023-02-05	Human Resources	HR Manager	75000.00	2023-08-19
David	Lee	david.lee@example.com	5553456789	2023-04-25	Finance	Finance Analyst	65000.00	2023-08-19

Σχήμα 22: Στιγμιότυπο πίνακα βάσης δεδομένων

Ο σκοπός της υλοποίησης είναι η καταγραφή των αλλαγών στη βάση αυτή και η ενημέρωση εάν παρατηρηθούν ύποπτες ενέργειες. Οι ενέργειες που θεωρούμε ύποπτες είναι, μεταξύ άλλων που θα αναλυθούν αργότερα, η αλλαγή του μισθού χωρίς να ενημερωθεί η ημερομηνία αλλαγής μισθού. Αν και μια τέτοια αλλαγή μπορεί να είναι για την διόρθωση λάθους, την ορίζουμε σαν ύποπτη καθώς μια πιθανή επίθεση μπορεί να είναι η αύξηση του μισθού χωρίς να φαίνεται πως αυτή έγινε πρόσφατα, δηλαδή ότι προϋπήρχε. Αυτό θα είχε ως σκοπό πιθανώς την παράλειψη ελέγχου από το λογιστήριο καθώς φαινομενικά δεν πρόκειται για καινούργια αλλαγή, συνεπώς μπορεί να θεωρηθεί πως αυτή έχει ελεγχθεί και επικυρωθεί στο παρελθόν. Για αυτό το λόγω όλες οι αλλαγές που γίνονται στη καταγράφονται έτσι ώστε να διαπιστωθεί οποιαδήποτε παραβίαση των κανόνων που θέτουμε τόσο σε πραγματικό χρόνο, όσο και μετά το πέρας της επίθεσης.

4.2 Τεχνολογίες

Παρακάτω θα γίνει μια σύντομη αναφορά στις τεχνολογίες που χρησιμοποιήθηκαν για σχεδίαση και υλοποίηση του περιβάλλοντος, της βάσης και του blockchain.

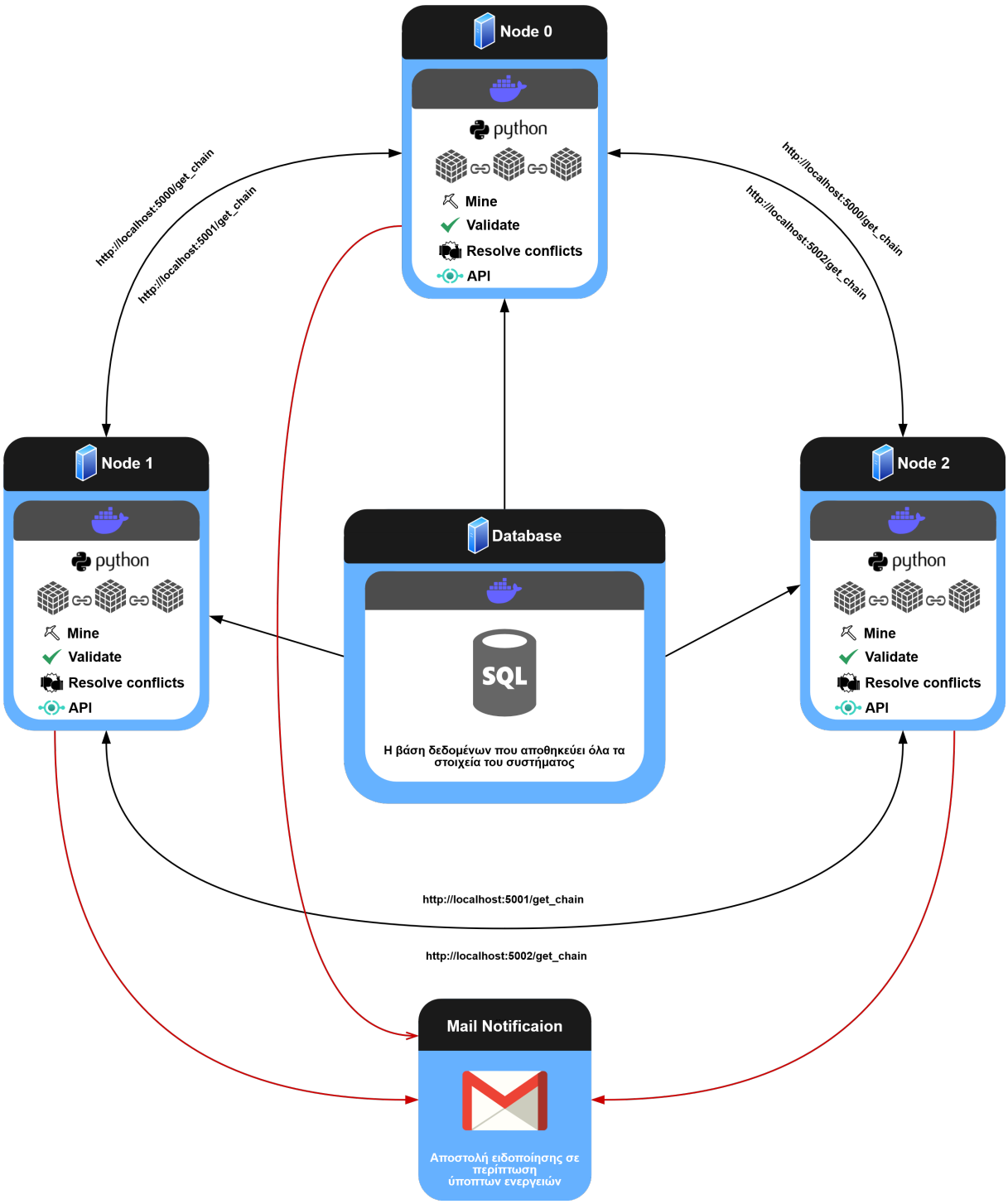
- Ως γλώσσα ανάπτυξης ολόκληρης της υλοποίησης αλλά και του blockchain επιλέχθηκε η Python, μια προγραμματιστική γλώσσα υψηλού επιπέδου. Δημιουργήθηκε αρχικά τον δεκαετία του 1980 από τον Guido van Rossum και έκτοτε έχει αναπτυχθεί σε μια ισχυρή και ευέλικτη γλώσσα που διαθέτει μια ευρεία γκάμα βιβλιοθηκών, που διευκολύνουν την επέκταση των δυνατοτήτων της, κάνοντας ιδανική για χρήση σε πολλούς τομείς. Για την ανάπτυξη της εφαρμογής δεν έγινε χρήση κάποιας έτοιμης βιβλιοθήκης (για το κομμάτι του blockchain), αντ'αυτού προτιμήθηκε να προγραμματιστούν όλα τα κομμάτια από το μηδέν, για την καλύτερη κατανόηση της τεχνολογίας και την δημιουργία εξειδικευμένης λύσης για το συγκεκριμένο πρόβλημα.

- Για την βάση χρησιμοποιήθηκε η MySQL, ένα ανοιχτού κώδικα σύστημα διαχείρισης βάσεων δεδομένων (DBMS) που χρησιμοποιείται ευρέως για την αποθήκευση, τη διαχείριση και την ανάκτηση δεδομένων. Δημιουργήθηκε αρχικά από τους Michael Widenius και David Axmark το 1995 και αργότερα αποκτήθηκε από την Oracle Corporation. Ωστόσο, η MySQL παραμένει διαθέσιμη ως ανοιχτού κώδικα λογισμικό υπό την άδεια GNU General Public License (GPL).

- Για περιβάλλον ανάπτυξης και εκτέλεσης επιλέχθηκε το Docker, μια πλατφόρμα λογισμικού που χρησιμοποιείται για τη δημιουργία, τη διανομή και την εκτέλεση εφαρμογών σε περιβάλλοντα που ονομάζονται "containers" (δοχεία). Τα containers είναι απομονωμένα περιβάλλοντα που περιλαμβάνουν όλα τα απαραίτητα εργαλεία, βιβλιοθήκες και παραμετροποιήσεις που απαιτούνται για την εκτέλεση μιας εφαρμογής. Προσφέρουν ένα απομονωμένο περιβάλλον στην εφαρμογή, επιτρέποντας την εύκολη εκτέλεση της ακόμα και σε διαφορετικά συστήματα χωρίς προβλήματα. Επιλέχθηκε να χρησιμοποιηθεί λόγω της ευκολίας που προσφέρει στην παραμετροποίηση της βάσης και της δυνατότητας δημιουργίας πολλαπλών container κάνοντας πιο εύκολη τόσο την ανάπτυξη της εφαρμογής όσο και την προσομοίωση της δικτυακής τοπολογίας του συστήματος. Έτσι κάθε κομμάτι μπορεί να τρέχει σε δικό του περιβάλλον container όπως θα γινόταν πιθανώς και σε μια πραγματική εφαρμογή (server).

4.3 Σχεδίαση

Στο παρακάτω διάγραμμα απεικονίζεται απλοϊκά η σχεδίαση και η λειτουργία της υλοποιήσεις που αναπτύχθηκε. Ολόκληρη η υλοποίηση τρέχει μέσα σε Docker με παραμετροποίηση τέτοια ώστε κατα την αρχική εκτέλεση, η βάση δεδομένων να αρχικοποιείται με δεδομένα, triggers και να δημιουργεί πολλαπλούς χρήστες ενώ οι κόμβοι λαμβάνουν τις απαραίτητες ρυθμίσεις δικτύου για την μεταξύ τους επικοινωνία, τους κωδικούς για την σύνδεση στη βάση και την αυτόματη εκτέλεση του blockchain.



Σχήμα 23: Διάγραμμα υλοποίησης

Παρατηρούμε στο κέντρο του διαγράμματος την βάση δεδομένων στην οποία γίνονται οι αλλαγές. Αυτές οι αλλαγές καταγράφονται από όλα τα τα Node καθένα από τα οποία διατηρεί δικιά του αλυσίδα. Όταν κάποιο node δημιουργήσει το καινούργιο block, αυτό λαμβάνεται και από τα υπόλοιπα και εφόσον δεν περιέχει λάθη, προστίθεται στην αλυσίδα. Με αυτή την σχεδίαση, ακόμα και αν υπάρξει πρόβλημα με ένα από τα nodes η διαδικασία καταγραφής θα συνεχίσει να δουλεύει ασταμάτητα, ενώ επιπλέον μπορούν να προστεθούν περισσότεροι κόμβοι.

Κάθε κόμβος πρέπει να είναι σε θέση να λαμβάνει τις αλλαγές της βάσης δεδομένων, να δημιουργεί και να διαχειρίζεται την αλυσίδα αλλά και να επικοινωνεί με τους υπόλοιπους κόμβους που απαρτίζουν το δίκτυο, έτσι ήταν απαραίτητη η χρήση κάποιας τεχνικής παραλλήλου υπολογισμού. Για αυτό το λόγο το πρόγραμμα που αναπτύχθηκε κάνει χρήση threads, δηλαδή μικρά ανεξάρτητα νήματα εκτέλεσης καθένα με τη δικιά του ανεξάρτητη στοίβα που εκτελούνται παράλληλα με τα υπόλοιπα νήματα που τρέχουν. Αυτό επιτρέπει την άμεση επικοινωνία μεταξύ των κόμβων αλλά και με τη βάση χωρίς καθυστερήσεις.

Όλες οι αλλαγές της βάσης, όχι μόνο του πίνακα, ακόμα και διαχειριστικές όπως για παράδειγμα η επανεκκίνηση της βάσης, η σύνδεση κάποιου χρήστη, η δημιουργία χρήστη, καταγράφονται. Τα δεδομένα που συγκρατούνται είναι η εντολή ή το ερώτημα (query ή command) ο χρήστης που το εκτέλεσε και η ώρα που εκτελέστηκε.


```
[
    {
        "index": 0,
        "previous_hash": "0",
        "timestamp": 1696852374,
        "last_data_timestamp": "1970-01-01 00:00:00",
        "data": "Genesis Block",
        "nonce": 0,
        "time_taken": 0,
        "pow": 0,
        "hash": "93d5299b53dc9943de7b596bb5abaed3b60335338052b87e27813012b1528de3"
    },


    {
        "index": 1,
        "previous_hash": "93d5299b53dc9943de7b596bb5abaed3b60335338052b87e27813012b1528de3",
        "timestamp": 1696852403,
        "last_data_timestamp": "2023-5-09 11:53:00.612769",
        "data": [
            {
                "event_time": "2023-10-09 11:53:00.253279",
                "user_host": "root[root] @ localhost []",
                "command_type": "Query",
                "argument": "SELECT * FROM company.salaries;"
            }
        ],
        "nonce": 139522,
        "time_taken": 8.234982967376709,
        "pow": "000082ed2dfae5f2c57b380d78ab07ba01bf3d4f05139c2a070e5330807701a5",
        "hash": "d54f5b510774c25da7e89e4fb0aec6a7e1f05a9167aceb7d4907c174e64c6e81"
    }
]
```


Παραπάνω φαίνεται η μορφή της αλυσίδας που συγκρατεί η υλοποίηση, περιέχει τα blocks καθένα απ’τα οποία διατηρεί δεδομένα (data), το αποτέλεσμα της συνάρτησης κατακερματισμού του ίδιου του block (hash), έναν αριθμό (index) που αντιπροσωπεύει τη θέση του, την χρονοσήμανση της τελευταίας φοράς που έλαβε δεδομένα από την βάση (last_data_timestamp), ένα αριθμό (noonce) που χρησιμοποιείται για την εύρεση της συνάρτησης κατακερματισμού (pow) με την δυσκολία που ζητάμε, το hash του προηγούμενου block, το χρόνο που έκανε να δημιουργηθεί το ίδιο το block και τέλος η χρονοσήμανση (timestamp) ένταξης του block στην αλυσίδα.

Στην παραπάνω αλυσίδα το πεδίο data του πρώτου block περιέχει το λεκτικό “Genessis block”, πρόκειται δηλαδή για το πρώτο block της αλυσίδας, για αυτό και έχει index, nonce, time_taken και pow ίσα

με το 0. Επίσης η τελευταία φορά που έλαβε δεδομένα από τη βάση αρχικοποιείται στην αρχή του epox time. Το block αυτό δημιουργείται αυτόματα μέσα στο node αν διαπιστωθεί πως δεν υπάρχει αλυσίδα. Το δεύτερο block της αλυσίδας περιέχει στο πεδίο data την ώρα εκτέλεσης (event_time) που ο χρήστης root συνδεδεμένος τοπικά στο server της βάσης (user_host) εκτέλεσε το ερώτημα (command_type = Query) SELECT * FROM company.salaries.

Για την επικοινωνία κάθε κόμβου με τους υπόλοιπους έγινε χρήση του Flask, ένα ελαφρύ framework για ανάπτυξη διαδικτυακών εφαρμογών στη Python, προκειμένου να δημιουργηθούν API endpoints σε κάθε node επιτρέποντας την μεταξύ τους επικοινωνία. Έτσι με ένα απλό αίτημα (request) στην διεύθυνση του node (πχ. http://127.0.0.1:5000/get_chain) λαμβάνουμε ως απάντηση την αλυσίδα που διατηρεί ο συγκεκριμένος κόμβος σε μορφή json.


```
"chain": [
  {
    "data": "Genesis Block",
    "hash": "698853e9408ce8f3bd91d777607acc0340448b680152271429f6a978433e8a5a",
    "index": 0,
    "last_data_timestamp": "1970-01-01 00:00:00",
    "nonce": 0,
    "pow": 0,
    "previous_hash": "0",
    "time_taken": 0,
    "timestamp": 1692544983
  }
]
```


Η μορφή της αλυσίδας που διατηρεί κάθε κόμβος, για την καλύτερη κατανόηση και αναγνωσιμότητα από τον άνθρωπο αλλά και για την ευκολία στη μεταφορά της ανάμεσα στους κόμβους είναι σε json (JavaScript Object Notation) μορφοποίηση. Κάθε node διατηρεί τοπικά ένα αρχείο json σαν μια non-SQL βάση δεδομένων της αλυσίδας, έτσι ακόμα και αν διακοπεί η λειτουργιά του node, δεν χάνονται τα δεδομένα που διατηρούσε. Για το λόγο αυτό υπάρχουν μέθοδοι μέσα στο πρόγραμμα που αναπτύχθηκε που μετατρέπουν από json σε blockchain αντικείμενα και το αντίθετο άλλα δεν υπάρχει λόγος να αναλυθούν περαιτέρω καθώς είναι απλή και ξεκάθαρη η λειτουργιά τους. Όταν το αντικείμενο Blockchain που διατηρεί την αλυσίδα, κατά την δημιουργία του, αρχικοποιεί την αλυσίδα, αν βρεθεί πως υπάρχει αυτό το backup αρχείο, αρχικοποιεί την αλυσίδα με αυτό, διαφορετικά δημιουργεί ένα καινούργιο Genesis Block. Το αντικείμενο αυτό διατηρεί ουσιαστικά μια λίστα από αντικείμενα τύπου block. Επίσης κάθε κόμβος διατηρεί λίστα received_blockchains, με τις αλυσίδες των υπολοίπων μέσα στο δίκτυο κόμβων. Αν και η αλυσίδα είναι σε μορφή json για την καλύτερη ανάγνωση, αποθήκευση και μεταφορά της, δεν είναι σε αυτή τη μορφή μέσα στο node. Για την δημιουργία, διατήρηση και επεξεργασία της αλυσίδας χρησιμοποιήθηκε αντικειμενοστρεφής προγραμματισμός.

Για να ελέγχουν τυχόν διάφορες στις αλυσίδες που διατηρεί κάθε κόμβος, πρώτα πρέπει να λάβει την αλυσίδα όλων τον υπολοίπων μέσα στο δίκτυο. Έτσι μετά τη δημιουργία καινούργιου block ζητά από τα υπόλοιπα nodes, μέσω του API που αναπτύχθηκε, την αλυσίδα τους. Έστω το σενάριο ότι καθένα από τα τρία node του δικτύου σε μια x χρονική στιγμή διατηρεί δικιά του μοναδική λίστα με n blocks. Όταν τελειώσει το πρώτο node, έστω το node_0, ζητάει τη λίστα των άλλων δυο, εφόσον τελείωσε πρώτο, η δικιά του λίστα έχει n+1 block, ενώ τα υπόλοιπα έχουν n block. Συνεπώς κρατάει τη δικιά του λίστα. Όταν τελειώσουν τα υπόλοιπα με τον υπολογισμού του επόμενου block, ζητάνε και αυτά με τη σειρά τους την

αλυσίδα από τα υπόλοιπα node μέσα στο δίκτυο, όταν λάβουν τη λίστα του node_0 υπολογίζουν πως έχουν τον ίδιο αριθμό blocks (n+1) σε αυτή την περίπτωση επιλεγούν να διατηρήσουν την αρχαιότερη αλυσίδα. Με αυτό το τρόπο όλο το δίκτυο των node διατηρεί μια ενιαία αλυσίδα.

Συνεπώς υπάρχουν δυο κανόνες για κάθε node:

- Αν η αλυσίδα σου είναι μεγαλύτερη από κάθε άλλη άμεσα στο δίκτυο, την διατηρείς.
- Αν η αλυσίδα σου είναι ίση μια κάποια άλλη μέσα στο δίκτυο, κράτησε την αρχαιότερη.

Για να δημιουργηθεί ένα block χρειάζεται ένας αλγόριθμος, για τις ανάγκες της εργασίας και με γνώμονα πως αυτή η αλυσίδα πρόκειται να τρέξει μέσα στο δίκτυο ενός ιδιωτικού οργανισμού, αναπτύχθηκε ένας συνδυασμός του PoW και του PoET. Σκοπός ήταν να μειωθούν οι δαπάνες υπολογιστικής ισχύος και συνεπώς ενέργειας. Έτσι ορίζουμε ένα ελάχιστο χρόνο αναμονής που κάθε node πρέπει να παραμείνει ανενεργό, όταν τελειώσει αυτός ο χρόνος ξεκινάει ο υπολογισμός του υποψηφίου block. Άρα θεωρητικά δημιουργείται ένα καινούργιο κάθε:

χρόνος αναμονής + *χρόνος υπολογισμού συνάρτησης κατακερματισμού*

Έτσι αν ορίσουμε πως θέλουμε να δημιουργούνται καινούργια block κάθε πέντε λεπτά, δεν σπαταλάμε ενεργεία στον υπολογισμό μιας συνάτησης κατακερματισμού που χρειάζεται πέντε λεπτά για να υπολογιστεί, περιμένουμε λίγα λεπτά (timeout period) και μετά ξεκινάμε τον υπολογισμό του hash.

Τέλος ελέγχονται τα καταγεγραμμένα στοιχεία, και αν αυτά κριθούν πως είναι ύποπτα αποστέλλονται ειδοποιήσεις μέσω ηλεκτρονικής αλληλογραφίας (email). Η υλοποίηση δεν λαμβάνει αντίμετρα για τις ύποπτες αυτές ενέργειες καθώς είναι επιτρεπτές βάσει των δικαιωμάτων που έχει ο χρήστης.

4.5 Παρουσίαση και επεξήγηση κώδικα

Παρακάτω θα γίνει μια επισκόπηση των αρχείων και των σημαντικών σημείων του κώδικα που αναπτύχθηκε.

- **Docker-compose.yml**

Πρόκειται για το αρχείο ρύθμισης (configuration file) που χρησιμοποιείται για τον καθορισμό και τη διαχείριση πολλαπλών containers. Η συγκεκριμένη μορφή αρχείου επιτρέπει τον ορισμό των διαφόρων περιβαλλόντων εκτέλεσης, των εικόνων Docker (images), και των παραμέτρων που απαιτούνται για τη διασύνδεση τους. Έτσι ορίζουμε 2 περιβάλλοντα για την υλοποίηση του blockchain το node_1 και το node_2. Το node_0 εκτελείται, εάν θελουμε, τοπικά δηλαδή έξω από το περιβάλλον του docker. Ωστόσο τίποτα δεν μας εμποδίζει από το να προσθέσουμε περισσότερα node μέσα στο docker.

Η παραμετροποίηση network_mode: host επιτρέπει την ευκολότερη επικοινωνία των Node που τρέχουν μέσα στο docker με αυτό που τρέχει εκτός απο αυτό.

```
version: '3.4'

services:

  node_1:
    container_name: "node_1"
    image: blockchain
    build: .
    network_mode: host
    environment:
      - port=5001
    depends_on:
      - sqldb


  node_2:
    container_name: "node_2"
    image: blockchain
    build: .
    network_mode: host
    environment:
      - port=5002
    depends_on:
      - sqldb


  sqldb:
    image: mysql
    container_name: "sqldb"
    command:  --default-authentication-plugin=mysql_native_password
    restart: always
    env_file: .env
    environment:
      MYSQL_ROOT_PASSWORD:  ${MYSQL_ROOT_PASSWORD}
      MYSQL_DATABASE:       ${MYSQL_DATABASE}
      MYSQL_USER:           ${MYSQL_USER}
      MYSQL_PASSWORD:       ${MYSQL_PASSWORD}
    volumes:
      - ./mysql-config/initialize_database:/docker-entrypoint-initdb.d
      - ./mysql-config/my.cnf:/etc/mysql/conf.d/my.cnf
      - database_mysql:/var/lib/mysql
      - ./mysql-config/logs:/var/log/mysql
    ports:
      - 3306:3306
```

Το τρίτο και τελευταίο container που δημιουργούμε είναι αυτό της βάσης δεδομένων στην οποία δημιουργούμε χρήστη root και user. Επίσης περνάμε ένα .env αρχείο που περιέχει όλους τους κωδικούς που θα χρειαστούμε. Τελος με τη χρήση των volumes αντιγράφουμε το αρχείο my.cnf καθώς και αρχεία στο docker-entrypoint-initdb.d τα οποία θα εκτελεστούν αυτόματα κατα τη δημιουργία του container.

- **mysql-config**

Όλα τα αρχεία που περιέχονται σε αυτό τον φάκελο αντιγράφονται μέσα στο container της βάσης κατα τη δημιουργία της, και έχουν σκοπό την αρχική παραμετροποίηση της.

- my.cnf

Ορίζει ως προορισμό για τα logs που κρατάει η βάση έναν πίνακα και τον κατάλογο που περιγράφεται στο general_log_file

```
[mysqld]
log_output = 'TABLE,FILE'
general_log_file = '/var/log/mysql/mysql.log'
```

- script.sh

Δημιουργεί επιπλέον χρήστες με τα απαραίτητα δικαιώματα

```
mysql -u root -p${MYSQL_ROOT_PASSWORD} -e "SET global general_log = 1; CREATE USER "$
{MYSQL_TEST_USER}"@'localhost' IDENTIFIED BY 'password';"

mysql -u root -p${MYSQL_ROOT_PASSWORD} -e "CREATE USER ${MYSQL_REGULAR_USER}@'%' IDENTIFIED
BY '${MYSQL_REGULAR_PASSWORD}';
GRANT SELECT, INSERT, UPDATE ON company.* TO ${MYSQL_REGULAR_USER}@'%';"
```

- init_script.sh

Δημιουργεί τη βάση, τον πίνακα και κάποια triggers που ενημερώνουν αυτόματα το πεδίο ημερομηνίας μισθού με την ημερομηνία αλλαγής του μισθού.

```
CREATE TABLE salaries (
    FIRST_NAME VARCHAR(50) DEFAULT NULL,
    LAST_NAME VARCHAR(50) NOT NULL,
    EMAIL VARCHAR(50) NOT NULL,
    PHONE_NUMBER VARCHAR(10) DEFAULT NULL,
    HIRE_DATE DATE NOT NULL,
    DEPARTMENT VARCHAR(50) NOT NULL,
    JOB VARCHAR(50) NOT NULL,
    SALARY DECIMAL(8,2) DEFAULT NULL,
    SALARY_DATE DATE NOT NULL
);

DELIMITER //
CREATE TRIGGER no_update_salary_date
BEFORE UPDATE ON salaries
FOR EACH ROW
BEGIN
    IF NEW.SALARY_DATE <> CURDATE() THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'MANUALLY UPDATING SALARY DATE IS NOT ALLOWED';
    END IF;
END;

CREATE TRIGGER update_salary_date
BEFORE UPDATE ON salaries
FOR EACH ROW
```

```
BEGIN
    IF NEW.SALARY <> OLD.SALARY THEN
        SET NEW.SALARY_DATE = CURDATE();
    END IF;
END;

CREATE TRIGGER insert_set_salary_date
BEFORE INSERT ON salaries
FOR EACH ROW
BEGIN
    SET NEW.SALARY_DATE = CURDATE();
END;
//
DELIMITER ;

SET global general_log = 1;
GRANT SELECT, INSERT, FILE, REFERENCES, SHOW DATABASES, SUPER ON *.* TO `blckchn`@`%` WITH
GRANT OPTION;
```

- **.env**

Ορίζει τις μεταβλητές που χρησιμοποιούνται τόσο από τα nodes όσο και από τη βάση. Έτσι αλλάζοντας τους κωδικούς από εδώ, ενημερώνονται αυτόματα σε όλα τα κομμάτια της εφαρμογής.

```
MYSQL_ROOT_PASSWORD='admin'
MYSQL_DATABASE='company'
MYSQL_USER='blckchn'
MYSQL_PASSWORD='secrete'
MYSQL_TEST_USER='testtest'
MYSQL_PORT='3306'
MYSQL_IP='127.0.0.1'
MYSQL_REGULAR_USER='ruser'
MYSQL_REGULAR_PASSWORD='ruserpwrd'
port=5000
```

- **Refresh.sh**

Ένα script το οποίο δημιουργήθηκε έτσι ώστε να διαγράφει τα απαραίτητα αρχεία μετά από κάθε εκτέλεση, προκειμένου να μπορεί να τρέξει εκ νέου η υλοποίηση όπως την πρώτη φορά.

```
#!/bin/bash

# Run with sudo
# Need to run chmod +x Refresh.sh

docker compose -f "docker-compose.yml" down

rm -r ./blockchain_logs/database/*
rm ./blockchain_logs/blockchain.json
rm mysql-config/logs/mysql.log

# Kill program running on port 5000
lsof -ti :5000 | xargs kill -9

sudo chmod 644 mysql-config/my.cnf

docker volume prune -a -f
docker image rm $(docker image ls -aq)

docker container ls

#docker compose -f "docker-compose.yml" up -d --build
```

- **query.txt**

Περιέχει δεδομένα σε μορφή κειμένου που χρησιμοποιούνται από το database_usage.sh που περιγράφεται παρακάτω.

- **requirements.txt**

Περιέχει όλες τις απαραίτητες βιβλιοθήκες που απαιτεί η υλοποίηση για την εκτέλεση της. Αυτές εγκαθίστανται αυτόματα κατα την δημιουργία του container.

- **blockchain_logs/nodes.txt**

Περιέχει την λίστα με όλα τα διαθέσιμα nodes.

http://localhost:5000

http://localhost:5001

http://localhost:5002

- **database_usage.sh**

Script το οποίο τροποποιεί τα δεδομένα του πίνακα, προσομοιώνοντας "νόμιμες" αλλαγές που θα είχε μια πραγματική βάση. Τα δεδομένα τα αντλεί από το αρχείο query.txt.

```
# Set enviroment variables
source ./.env

# Database credentials
DB_HOST=$MYSQL_IP
DB_PORT=$MYSQL_PORT
DB_NAME=$MYSQL_DATABASE
DB_USER=$MYSQL_REGULAR_USER
DB_PASSWORD=$MYSQL_REGULAR_PASSWORD
DB_TABLE_NAME='salaries'

# File containing records to insert
records_file="queries.txt"

# Pause duration in seconds
pause_duration=5

# Loop through records in the file and insert into the database
while IFS= read -r record
do
    # Split the record into individual fields
    IFS=',' read -ra fields <<< "$record"

    # Extract the fields
    FIRST_NAME="${fields[0]}"
    LAST_NAME="${fields[1]}"
    EMAIL="${fields[2]}"
    PHONE_NUMBER="${fields[3]}"
    HIRE_DATE="${fields[4]}"
    DEPARTMENT="${fields[5]}"
    JOB="${fields[6]}"
    SALARY="${fields[7]}"
    SALARY_DATE="${fields[8]}"

    record_exists=$(mysql -h $DB_HOST -P $DB_PORT -u $DB_USER -p$DB_PASSWORD -D $DB_NAME -se "SELECT
COUNT(*) FROM $DB_TABLE_NAME WHERE EMAIL='${fields[2]}'")

    if [ "$record_exists" -eq 0 ]; then
        # SQL query to insert the record into the table
        query="INSERT INTO $DB_TABLE_NAME (FIRST_NAME, LAST_NAME, EMAIL, PHONE_NUMBER, HIRE_DATE,
DEPARTMENT, JOB, SALARY, SALARY_DATE) VALUES
        ('$FIRST_NAME', '$LAST_NAME', '$EMAIL', '$PHONE_NUMBER', '$HIRE_DATE', '$DEPARTMENT',
'$JOB', $SALARY, $SALARY_DATE);"
    else
        # Update record
        query="UPDATE $DB_TABLE_NAME SET FIRST_NAME='$FIRST_NAME', LAST_NAME='$LAST_NAME',
EMAIL='$EMAIL', PHONE_NUMBER='$PHONE_NUMBER', HIRE_DATE='$HIRE_DATE',
        DEPARTMENT='$DEPARTMENT', JOB='$JOB', SALARY=$SALARY, SALARY_DATE=$SALARY_DATE WHERE
EMAIL='$EMAIL'"
```

```
    fi
    echo $query
    # Execute the SQL query using mysql command
    mysql -h $DB_HOST -P $DB_PORT -u $DB_USER -p$DB_PASSWORD -D $DB_NAME -e "$query"

    # Pause for specified duration
    sleep $pause_duration
done < "$records_file"
```

- **Κλάση Blockchain και Block**

Καθένα από τους κόμβους διατηρεί ένα δικό του αντικείμενο blockchain. Αυτό είναι υπεύθυνο για την αρχικοποίηση και διαχείρηση της αλυσίδας.

class Blockchain:

```
    def __init__(self):
        self.chain = self.read_blockchain_from_json()
        self.nodes = self.register_nodes()
        self.received_blockchains = []

        if not self.chain:
            self.chain = [Block(0, "0", int(time.time()), "Genesis Block", 0, 0)]

        self.difficulty = 4
        self.desired_time = 5
```

Αυτό που διαχειρίζεται είναι ουσιαστικά μια λίστα από αντικείμενα τύπου Block. Τα δεδομένα που συγκρατεί κάθε block αναλύθηκαν παραπάνω.

```
class Block:
    def __init__(self, index, previous_hash, timestamp, data, nonce, time_taken, work=0,
last_data_timestamp='1970-01-01 00:00:00', hash=0):
        self.index = index
        self.previous_hash = previous_hash
        self.timestamp = timestamp
        self.last_data_timestamp = last_data_timestamp
        self.data = data
        self.nonce = nonce
        self.time_taken = time_taken
        self.pow = work
        if hash : self.hash = hash
        else: self.hash = self.calculate_hash()
```

Κατά τον υπολογισμό του block υπάρχει μια συνάρτηση κατακερματισμού που ζητάει να βρεθεί αποτέλεσμα με τέσσερα προπορευόμενα μηδενικά ψηφία (self.difficulty). Έτσι στο παράδειγμα που ακολουθεί ενός block που έτυχε να μην έχει δεδομένα, παρατηρούμε πως ο χρόνος που χρειάστηκε για τον υπολογισμό του ήταν πέντε δευτερόλεπτα συν το χρόνο που χρειάστηκε να υπολογιστεί το pow με τέσσερα προπορευόμενα μηδενικά, δηλαδή 0,15 δευτερόλεπτα. Το nonce είναι ο αυξανόμενος αριθμός που αλλάζει προκείμενου να πετύχουμε τον αριθμό των επιθυμητών μηδενικών ψηφίων στην αρχή του hash .


```
{
  "data": [],
  "hash": "c995adf112fafe1e66ecc6a5f35465095bde60778d9685ad2dbc690c8ba19f00",
  "index": 3,
  "last_data_timestamp": "2023-08-20 15:24:05.114420",
  "nonce": 733200,
  "pow": "0000ac7ce220addc504928fdf690745b82acebc357926239c10cf134d4aa2324",
  "previous_hash": "a032176b3556f0c101004245a37b4067320830567aa9903356dd9d05a2da9b2b",
  "time_taken": 5.0150229930877686,
  "timestamp": 1692545089
}
```

- **Συνάρτηση σύνδεσης με την βάση**

Κάθε node από την στιγμή που θα ξεκινήσει θα δημιουργήσει ένα genesis block εφ'όσον δεν έχει κάποια αλυσίδα, και στη συνέχεια θα περιμένει να συνδεθεί με την βάση δεδομένων.

```
def database_connect():
    start_time = time.time()
    while not is_mysql_running():
        print("Waiting for MySQL to start...")
        time.sleep(10)

    # Connect to the MySQL server
    connection = mysql.connector.connect(**config)
    cursor = connection.cursor()
    cursor.execute("SET GLOBAL general_log = 'ON'")
    return connection
```

Αν δεν είναι εφικτή η σύνδεση με τη βάση περιμένει και προσπαθεί ξανά. Αυτό είναι απαραίτητο καθώς όταν ξεκινάει το docker η αρχικοποίηση της βάσης χρειάζεται περισσότερο χρόνο από την αρχικοποίηση των node.

- **Συνάρτηση αυτόματης φόρτωσης μεταβλητών**

Κάθε node λαμβάνει τα απαραίτητα στοιχεία για την επικοινωνία του με τη βάση, όπως για παράδειγμα κωδικούς από το αρχείο .env όπως αναλύθηκε σε προηγούμενη υποενότητα

```
def load_env_variable(env_var_name):
    """
    Loads variables from .env file only if the corresponding enviroment
    variable doesn't already exist
    """
    if os.environ.get(env_var_name) is None:
        if os.path.exists(".env"):
            load_dotenv(".env")
        else:
            print("Error .env file not found")

nodes_file_path = "./blockchain_logs/nodes.txt"
load_env_variable("port")
port = os.environ.get('port')
print(f'HI FROM THE DOCKER THING MY PORT IS: {port}')

load_env_variable("MYSQL_USER")
load_env_variable("MYSQL_PASSWORD")
load_env_variable("MYSQL_IP")
load_env_variable("MYSQL_DATABASE")

config = {
    'user': os.getenv("MYSQL_USER"),
    'password': os.getenv("MYSQL_PASSWORD"),
    'host': os.environ.get('MYSQL_IP'),
    'port' : '3306',
    'database': os.getenv("MYSQL_DATABASE")
}
```

- **Συνάρτηση δημιουργίας block**

Η κυρία διαδικασία κάθε node είναι να δημιουργεί blocks, αυτή τρέχει σε δικό της thread, και ονομάζεται mine_block().

```
    def mine_blocks(self):
      while True:
          last_block = self.chain[-1]
          data = get_data_payload(last_block.last_data_timestamp)
          new_block = self.proof_of_work(last_block, data)
```

```
        self.chain.append(new_block)
        self.save_block_to_json(new_block)
        print(f"Block {new_block.index} mined with nonce {new_block.nonce}")
        print(f"Block hash: {new_block.hash}")
        print(f"Elapsed time: {new_block.time_taken} seconds")

        code, invalid_block, validation_result = is_valid_chain(self.difficulty, self.chain)

        if code == 2:
            print(f"There is no blockchain to check yet...")
            print(f"{validation_result}")
            print(f"{invalid_block}")
        elif code == 1:
            print(f"The blockchain is NOT valid. Reason: {validation_result}")
            print(f"Invalid Block Index: {invalid_block.index}\n")
        else:
            print("The blockchain is valid.\n")

        self.resolve_conflicts()
```

Αυτή η μέθοδος τρέχει συνεχόμενα, συνδέεται εκ νέου στη βάση και λαμβάνει τα δεδομένα, τρέχει το παραμετροποιημένο proof of work αλγόριθμο, προσθέτει το καινούργιο block στην αλυσίδα, αποθηκεύει την αλυσίδα στο backup json αρχείο και ελέγχει την αλυσίδα που διατηρεί για λάθη. Τέλος ελέγχει για διαφορές ανάμεσα στα blockchain των υπολοίπων node με τη χρήση του resolve_conflicts. Για να ελεγχθεί η ακεραιότητα της αλυσίδας αναπτύχθηκε η is_valid_chain(). Αν η αλυσίδα δεν είναι κενή ή έχει μόνο ένα block (Genesis block), ελέγχει τις συναρτήσεις κατακερματισμού

```
        # Verify block integrity
        if block.hash != block.calculate_hash():
            return 1, block, "Integrity check failed"

        # Verify chain consistency
        if block.previous_hash != previous_block.hash:
            return 1, block, "Chain consistency check failed"

        # Verify Proof of Work
        if block.pow[:difficulty] != "0" * difficulty:
            return 1, block, "Proof of Work check failed"
```

- **Συνάρτηση συγχρονισμού blockchain**

Η συνάρτηση που λαμβάνει τις αλυσίδες από όλους τους κόμβους, τις συγκρίνει και επιλέγει ποια θα διατηρηθεί.

```
def resolve_conflicts(self):
        # Request other node blockchains
        blockchain.received_blockchains.clear()
        print(f"Requesting blockchain from: {self.nodes}")

        received_blockchains = self.get_blockchain()

        nodes_port = list(received_blockchains.keys())

        print(f"RECEIVED BLOCKCHAINS FROM NODES: {nodes_port}")

        for block in self.chain:
            if block.index == 0:
                oldest_block = block.timestamp

        longest_chain_length = len(blockchain.chain)
        longest_chain_key = None
        for nodes_blockchains, inner_data in received_blockchains.items():
            if len(inner_data['chain']) == longest_chain_length:
                #keep the oldest blockchain
                if oldest_block > inner_data['chain'][0]['timestamp']:
                    longest_chain_key = nodes_blockchains
            elif len(inner_data['chain'])> longest_chain_length:
                longest_chain_length = len(inner_data['chain'])
                longest_chain_key = nodes_blockchains
```

Τέλος κάθε node πριν δημιουργήσει ένα block θα πρέπει να αναζητήσει τις διαφορές που έχουν γίνει στη βάση από την τελευταία φορά που ελέγχθηκε. Αυτό γίνεται με την get_data_payload η οποία κάνοντας ερωτήματα στη βάση είναι σε θέση να συγκρίνει και να βρει διάφορες στον πινάκα. Εφόσον υπάρχουν προσπαθεί να λάβει όσο περισσότερες πληροφορίες γίνεται για αυτές και τις καταγράφει στο επόμενο block που θα δημιουργήσει.


```
"data": [
        {
          "argument": "GRANT SELECT, INSERT, FILE, REFERENCES, SHOW DATABASES, SUPER ON *.* TO
                      `blckchn`@`%` WITH GRANT OPTION",
          "command_type": "Query",
          "event_time": "2023-08-20 15:24:04.331682",
          "user_host": "root[root] @ localhost []"
        },
        {
          "argument": "",
          "command_type": "Quit",
          "event_time": "2023-08-20 15:24:04.536782",
          "user_host": "root[root] @ localhost []"
        }
      ]
```


Έτσι αποθηκεύονται δεδομένα μέσα στο data με την παραπάνω μορφή, ενεργεία (εντολή, query), τύπος εντολής, χρονοσήμανση εκτέλεσης και τέλος ο χρήστης που πραγματοποίησε τις ενέργειες.

- **Detection System**

Για την προειδοποίηση ύποπτων ενεργειών υπάρχει η alert(). Λαμβάνοντας δεδομένα από την αλυσίδα εξετάζει αν κάποια από τις ενέργειες που πραγματοποιήθηκαν είναι ύποπτες, σε αυτή την περίπτωση αποστέλλει προειδοποιήσεις. Ύποπτες ενέργειες θεωρούμε την διαγραφή της βάσης ή του πινάκα, την αλλαγή ημερομηνίας μισθού χειροκίνητα ή την αλλαγή μισθού χωρίς να έχει ενημερωθεί η ημερομηνία αλλαγής μισθού, την διαγραφή ή απενεργοποίηση trigger της βάσης που είναι υπεύθυνα για την αυτόματη αλλαγή ημερομηνίας μισθού, την απενεργοποίηση του συστήματος logging της mysql ή τον τερματισμό λειτουργίας της βάσης. Διαβάζοντας τις εντολές που εκτέλεσαν οι χρήστες στη βάση δεδομένων από την αλυσίδα είναι σε θέση, με την χρήση κανονικών εκφράσεων, να αναγνωρίσει μοτίβα, πρότυπα και λέξεις κλειδιά και να κρίνει αν πρόκειται για κάποια εντολή που παραβιάζει τους κανόνες που θέσαμε. Έτσι για παράδειγμα ορίζοντας ως πρότυπο ”\DROP\s+TRIGGER\b” ο αλγόριθμος είναι σε θέση αναγνωρίσει την εντολή DROP TRIGGER και να ενημερώσει.

4.6 Οδηγίες εγκατάστασης και εκτέλεσης.

Για την εκτέλεση της υλοποίησης αρκεί να υπάρχει εγκατεστημένο στο σύστημα μας το Docker, στη συνέχεια μεταφέροντας τα αρχεία του κώδικα και εκτελώντας την εντολή docker compose -f "docker-compose.yml" up -d —build, εφόσον δεν επιθυμούμε να αλλάξουμε κάποια από τις ρυθμίσεις, δημιουργούνται τρία docker containers το sql, το node_1 και το node_2. Τρέχοντας την εντολή docker ps μπορούμε να δούμε τα ενεργά containers που έχουμε.

```
CONTAINER ID   IMAGE
67d2aa11265e   blockchain
736a666df3d3   blockchain
cb036bbbf0b2   mysql
```

Σχήμα 24: Αποτελέσματα εντολής docker ps

Έτσι δημιουργείται ένα container για την SQL το οποίο αρχικοποιείται με ρυθμίσεις και δεδομένα αυτόματα και δυο container Blockchain τα οποία και αυτά, ομοίως, αρχικοποιούνται αυτόματα ξεκινώντας αμέσως την λειτουργία των node. Στη συνέχεια μπορούμε να τρέξουμε τοπικά, δηλαδή εκτός του docker, το τρίτο node αν το επιθυμούμε. Η υλοποίηση δεν έχει σχεδιαστεί για συγκεκριμένο αριθμό node μπορεί να τρέξει ακόμα και με ένα αν το επιθυμούμε. Για το τερματισμό της λειτουργίας του, εκτελούμε την εντολή docker compose -f "docker-compose.yml" down. Στη συνέχεια τρέχοντας εκ νέου την εντολή docker ps θα πρέπει να μας φέρει μηδέν αποτελέσματα. Λόγω του ότι έχει παραμετροποιηθεί το περιβάλλον του docker με τη χρήση Volumes να κρατάει τα δεδομένα μετά το τερματισμό της λειτουργίας των container (persistent data) αν επιθυμούμε να τρέξουμε τα container σαν να μην έχουν τρέξει στο παρελθόν, ίσως λόγω του ότι αλλάξαμε κάποια παραμετροποίηση που το απαιτεί, αν βρισκόμαστε σε περιβάλλον linux τρέχοντας με διαχειριστικά δικαιώματα (sudo) το bash αρχείο ./Refresh.sh θα διαγράψουμε τα απαραίτητα αρχεία και τα container θα δημιουργήσουν καινούργια όταν ξεκινήσουν. Σε άλλη περίπτωση θα πρέπει να δούμε πως διαγράφονται τα volumes και τα αρχεία στο λειτουργικό που χρησιμοποιούμε (host operating system).

Εάν το επιθυμούμε μπορούμε να τρέξουμε και το bash αρχείο database_usage.sh προκειμένου να έχουμε πραγματικά και νόμιμα δεδομένα και εντολές να τρέχουν μέσα στη βάση μας, κάνοντας πιο ρεαλιστικό το σενάριο της επίθεσης και πιο κοντά στην πραγματικότητα τα δεδομένα που καλείται να αποθηκεύσει το blockchain. Για τις εντολές της επίθεσης μπορούμε είτε να τις πραγματοποιήσουμε χειροκίνητα, κάνοντας σύνδεση στη βάση με διάφορους χρήστες, τρέχοντας εντολές και βλέποντας το αποτέλεσμα, είτε να τρέξουμε το αρχείο database_attack.sh το οποίο δημιουργήθηκε για να κάνει ακριβώς την ίδια δουλειά αυτοματοποιημένα.

Καθώς τρέχει η υλοποίηση μπορούμε να δούμε την αλυσίδα που διατηρεί κάθε node, απλά και μόνο ανοίγοντας ένα περιηγητή διαδικτύου (Browser) και πηγαίνοντας στις διευθύνσεις που τρέχει το καθένα, δηλαδή localhost ή 127.0.0.1, την πόρτα (port) στην οποία ακούει και ζητώντας το endpoint /get_chain. Συνεπώς όλη η διεύθυνση θα είναι για παράδειγμα http://localhost:5000/get_chain, ενώ το αποτέλεσμα θα είναι μια σελίδα με δεδομένα σε μορφή json.

4.7 Προσομοίωση επιθέσεων

Προκειμένου να ελεγχθεί η αποτελεσματικότητα και η χρηστικότητα της υλοποίησης είναι αναγκαίο να τρέξουμε ύποπτες ή κακόβουλες εντολές και ενέργειες στο σύστημα και να δούμε πως θα ανταποκριθεί. Για αυτό το λόγο θα γίνει χρήση μελέτης περιπτώσεων (case studies).

4.7.1 Τροποποίηση ημερομηνίας αλλαγής μισθού

Η χειροκίνητη τροποποίηση ημερομηνίας αλλαγής μισθού σε μια προγενέστερη. Αυτό μπορεί να συμβεί είτε λόγω λάθους, διόρθωσης λάθους ή κακόβουλων προθέσεων. Πρόκειται για επίθεση από χρήστη της βάσης που δεν χρειάζεται να έχει προχωρημένα δικαιώματα και δεν είναι απαραίτητο να έχει τεχνικές γνώσεις.

FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	DEPARTMENT	JOB	SALARY	SALARY_DATE
John	Doe	john.doe@example.com	5551234567	2023-01-15	Engineering	Software	60000.00	2023-08-01
Jane	Smith	jane.smith@example.com	5559876543	2022-11-20	Sales	Manager	80000.00	2023-08-01
Michael	Johnson	michael.johnson@example.com	5557890123	2023-03-10	Marketing	Specialist	55000.00	2023-08-01
Sarah	Williams	sarah.williams@example.com	5552345678	2023-02-05	Human Resources	HR Manager	75000.00	2023-08-01
David	Lee	david.lee@example.com	5553456789	2023-04-25	Finance	Finance Analyst	65000.00	2023-08-01

Σχήμα 25: Αρχική κατάσταση βάσης δεδομένων.

Έστω λοιπόν η παραπάνω κατάσταση στη βάση δεδομένων τη στιγμή εκτέλεσης της ύποπτης ενέργειας. Ο χρήστης, επιχειρεί να αλλάξει έναν μισθό. Την στιγμή της αλλαγής η ίδια η βάση θα τον ενημερώσει πως αυτό που επιχειρεί να κάνει δεν είναι επιτρεπτό.

```
mysql> UPDATE company.salaries SET SALARY = 80000, SALARY_DATE = '2023-08-15' WHERE FIRST_NAME='John' AND LAST_NAME='Doe';
ERROR 1644 (45000): Manual changes to the date field are not allowed.
```

Σχήμα 26: Αποτέλεσμα εκτέλεσης εντολής αλλαγής μισθού

Ενώ παράλληλα, βλέποντας την αλυσίδα (http://localhost:5000/get_chain) μπορούμε να δούμε πως ο χρήστης επιχείρησε να κάνει αλλαγή που δεν επιτρέπεται. Η οποία και σταμάτησε από την ίδια την SQL και δεν εκτελέστηκε ποτέ.


```
data:
  0:
    argument:         "UPDATE company.salaries SET SALARY = 80000, SALARY_DATE = '2023-08-15' WHERE FIRST_NAME='John' AND LAST_NAME='Doe'"
    command_type:     "Query"
    event_time:       "2023-08-20 20:42:54.015191"
    user_host:        "ruser[ruser] @  [172.18.0.1]"
  1:
    argument:         "SIGNAL SQLSTATE '45000'\n       SET MESSAGE_TEXT = 'Manual changes to the date field are not allowed.'"
    command_type:     "Query"
    event_time:       "2023-08-20 20:42:54.015537"
    user_host:        "root[root] @ localhost [localhost]"
hash:                 "bedb2020f7e80a7b5797e4ee6c69afe75853654417021d919d284d963033de23"
index:                111
last_data_timestamp:  "2023-08-20 20:42:54.015537"
nonce:                430638
pow:                  "000054ad726620ac6ea8a109e3ff85c820fba324ac7fd4c80060bb4c0dc954cf"
previous_hash:        "54155397b8cc921287dad16c163f1d69bb6dfbe03ecc30a9a0e8afdadd584c4f"
time_taken:           8.233756303787231
timestamp:            1692564187
```


Σχήμα 27: Εγγραφή block αλυσίδας.

Βλέπουμε την εντολή, το χρήστη, και την ώρα εκτέλεσης, καθώς και το μήνυμα που εμφάνισε το σύστημα. Αν και για κάτι τέτοιο δεν αποστέλλεται ειδοποίηση από τα node, καθώς η αλλαγή δεν πέρασε στη βάση, καταγράφεται κανονικά για μελλοντική αναφορά ή μελέτη.

4.7.2 Απενεργοποίηση sql triggers

Η απενεργοποίηση των sql triggers που αποτρέπουν την αλλαγή των ημερομηνιών και εν συνέχεια η χειροκίνητη αλλαγή της ημερομηνίας. Αυτή η ενέργεια απαιτεί διαχειριστικούς ή αυξημένων δικαιωμάτων λογαριασμούς. Καθώς αν εκτελεστεί από λογαριασμό απλού χρήστη όπως πριν, αφενός θα πρέπει ο επιτιθέμενος να γνωρίζει το όνομα του trigger που προσπαθεί να απενεργοποιήσει και αφετέρου, ακόμα και αν γνωρίζει το όνομα δεν θα του επιτραπεί να το απενεργοποιήσει

```
mysql> show triggers
    -> ;
Empty set (0.01 sec)
mysql> DROP TRIGGER IF EXISTS company.update_salary_date;
ERROR 1142 (42000): TRIGGER command denied to user 'ruser'@'172.27.0.1' for table 'salaries'
```

Σχήμα 28: Αποτελέσματα προσπάθειας απενεργοποίησης triggers με λογαριασμό περιορισμένων δικαιωμάτων

Ενώ όπως και πριν η προσπάθεια καταγράφεται από το blockchain σύστημα, αλλά αυτή την φορά αποστέλλεται προειδοποίηση η μορφή της οποίας παρουσιάζεται αργότερα.

```
▼ data:
  ▼ 0:
      argument:       "DROP TRIGGER IF EXISTS company.update_salary_date"
      command_type:   "Query"
      event_time:     "2023-08-20 22:13:04.014024"
      user_host:      "ruser[ruser] @  [172.27.0.1]"
```

Σχήμα 29: Καταγραφή ενεργειών από το σύστημα.

Αν η επίθεση γίνει με την χρήση διαχειριστικού ή αυξημένων δικαιωμάτων λογαριασμό και από άτομο με τεχνικές γνώσεις, τότε αρχικά θα απενεργοποιήσει το trigger

```
mysql> DROP TRIGGER IF EXISTS company.update_salary_date;
Query OK, 0 rows affected (0.16 sec)
```

Σχήμα 30: Απενεργοποίηση trigger

Θα αλλάξει τα δεδομένα που θέλει

```
mysql> UPDATE company.salaries SET SALARY=90000, SALARY_DATE = '2023-07-05' WHERE FIRST_NAME='John' AND LAST_NAME='Doe';
Query OK, 1 row affected (0.05 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

Σχήμα 31: Πραγματοποίηση μη επιτρεπτών ενεργειών με απενεργοποιημένα τα triggers

Και στη συνέχεια θα ξανά ενεργοποιήσει τα triggers, έτσι ώστε να μην γίνει αντιληπτή η αλλαγή

Όπως φαίνεται και στη βάση όλες οι αλλαγές πραγματοποιήθηκαν με επιτυχία

FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	DEPARTMENT	JOB	SALARY	SALARY_DATE
John	Doe	john.doe@example.com	5551234567	2023-01-15	Engineering	Software	90000.00	2023-07-05
Jane	Smith	jane.smith@example.com	5559876543	2022-11-20	Sales	Manager	80000.00	2023-08-01
Michael	Johnson	michael.johnson@example.com	5557890123	2023-03-10	Marketing	Specialist	55000.00	2023-08-01
Sarah	Williams	sarah.williams@example.com	5552345678	2023-02-05	Human Resources	HR Manager	75000.00	2023-08-01
David	Lee	david.lee@example.com	5553456789	2023-04-25	Finance	Finance Analyst	65000.00	2023-08-01

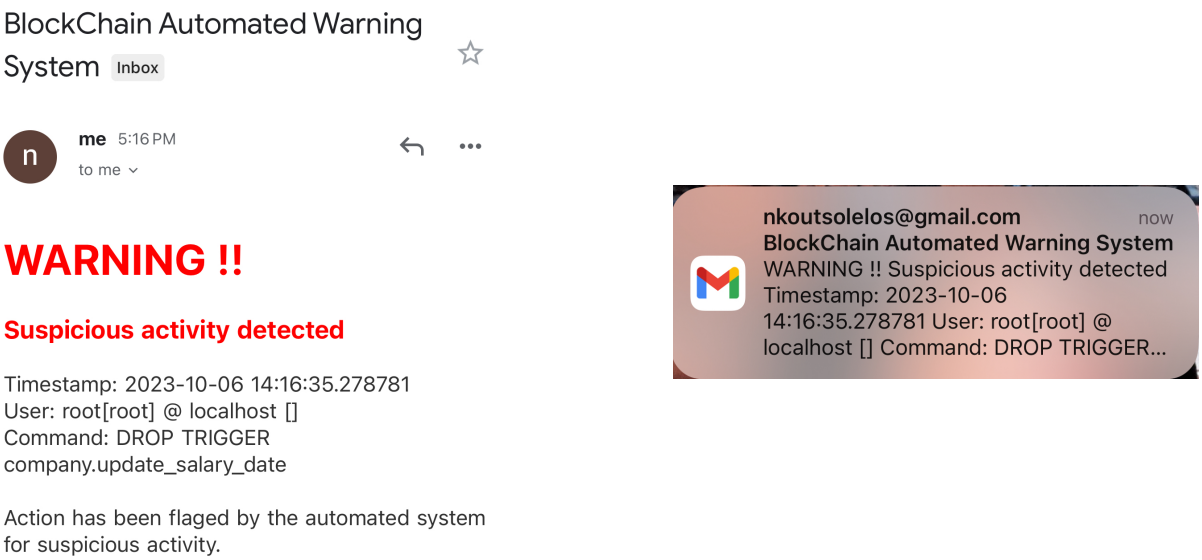
Σχήμα 32: Επιτυχημένες αλλαγές στη βάση.

Ενώ ζητώντας από κάποια από τα τρία node το blockchain


```
"data": [
    {
        "argument": "DROP TRIGGER IF EXISTS company.update_salary_date",
        "command_type": "Query",
        "event_time": "2023-08-20 22:28:24.716669",
        "user_host": "root[root] @ [172.27.0.1]"
    }
],
"hash": "bd5fbd2b4de777c1235fc2164f54008e8e43138cc4b9bd19157b97076856ad01",
"index": 195,
"last_data_timestamp": "2023-08-20 22:28:24.716669",
"nonce": 481096,
"pow": "0000f40b6a29596cd747c5aaf7d730bd09617111ccd8fa01d6c27901cd794421",
"previous_hash": "b52424585e6c0bc6ecdee531781b6b14e8eadb769f2135898c6259a27415002d",
"time_taken": 5.499082565307617,
"timestamp": 1692570511
},
{
"data": [
    {
        "argument": "UPDATE company.salaries SET SALARY=90000, SALARY_DATE = '2023-07-05' WHERE FIRST_NAME='John' AND LAST_NAME='Doe'",
        "command_type": "Query",
        "event_time": "2023-08-20 22:28:29.131208",
        "user_host": "root[root] @ [172.27.0.1]"
    }
],
"hash": "542520d379266d8d1f0b31cf23ee0825d9e16e9d47d6426ce8be78079fd80ee4",
"index": 196,
"last_data_timestamp": "2023-08-20 22:28:29.131208",
"nonce": 370815,
"pow": "00009f6385da8860ec17dc4985331b6be48379b8287f21ed9e5e9c75b1eab036",
"previous_hash": "bd5fbd2b4de777c1235fc2164f54008e8e43138cc4b9bd19157b97076856ad01",
"time_taken": 5.027125358581543,
"timestamp": 1692570516
},
{
"data": [
    {
        "argument": "CREATE TRIGGER update_salary_date\nBEFORE UPDATE ON salaries\nFOR EACH ROW\nBEGIN\n    IF NEW.SALARY <> OLD.SALARY
        "command_type": "Query",
        "event_time": "2023-08-20 22:28:33.773765",
        "user_host": "root[root] @ [172.27.0.1]"
    }
],
```


Σχήμα 33: Στιγμιότυπο blockchain μετά την επίθεση.

Παρατηρούμε πως βλέπουμε όλες τις εντολές που εκτελέστηκαν, το χρήστη που τις εκτέλεσε καθώς και την ώρα που εκτελέστηκαν. Όλα έχουν καταγράφει. Επίσης λόγω των κανόνων που θέσαμε στο σύστημα λαμβάνουμε προειδοποίηση.



Σχήμα 34: Προειδοποίηση συστήματος για την απενεργοποίηση trigger.

Δεν είναι απαραίτητο πως αυτές οι ενέργειες θα εκτελούνταν με κακόβουλο σκοπό, μπορεί ο λόγος που ήταν απαραίτητο να απενεργοποιηθεί το trigger προκειμένου να ενημερωθεί η ημερομηνία, να ήταν η διόρθωση κάποιου λάθους.

4.7.3 Απενεργοποίηση Node

Λόγω των δικαιωμάτων και των γνώσεων για το σύστημα, ο χρήστης μπορεί να γνωρίζει την ύπαρξη του blockchain και να προσπαθήσει να το απενεργοποιήσει. Αν υποθέσουμε πως έχει πρόσβαση στο σύστημα που τρέχει το node και μπορεί να το απενεργοποιήσει και στη συνέχεια να εκτελέσει την ίδια ακριβώς επίθεση τότε ο κόμβος που δέχτηκε την επίθεση, προφανώς και δεν θα απαντήσει στην περίπτωση που ζητήσουμε να μας αποστείλει την αλυσίδα. Ωστόσο ζητώντας την αλυσίδα από ένα εκ των δυο εναπομένων κόμβων θα λάβουμε απάντηση κανονικά. Αυτοί θα συνεχίσουν ανενόχλητοι την διαδικασία δημιουργίας καινούργιων block καταγράφοντας τις κινήσεις του. Όπως αναφέρθηκε και πριν το δίκτυο μπορεί να τρέξει ακόμα και με ένα κόμβο.

Σχήμα 35: Στιγμιότυπο blockchain των εναπομενόντων node.

Ένας τρόπος να επιτεθεί με επιτυχία και στο δίκτυο των κόμβων θα ήταν να δημιουργήσει ένα δικό του κόμβο, στον οποίο θα γέμιζε μια δικιά του νόμιμη αλλά ψεύτικη αλυσίδα, μεγαλύτερη σε έκταση από αυτή που τη διατηρούν τα υπόλοιπα node μέχρι εκείνη την χρονική στιγμή. Έτσι εκτελώντας την επίθεση του θα μπορούσε εν συνεχεία να θέσει το κακόβουλο node στο δίκτυο και όταν θα έρχονταν η ώρα να επιλεχθεί η αλυσίδα, αυτομάτως σύμφωνα με την τρέχουσα υλοποίηση θα νικούσε η δικιά του. Αυτή η επίθεση ενδέχεται να πετύχει στην αρχή της αλυσίδας, ωστόσο όσο προστίθενται blocks τόσο πιο χρονοβόρα και κοστοβόρα θα ήταν. Από κάποια στιγμή και μετά θα ήταν ρεαλιστικά αδύνατη.

Στο πλήθος των επιθέσεων που δοκιμάστηκαν, από διαφορετικούς χρήστες και με διαφορετικές τεχνικές το σύστημα κατάφερε πάντα να συγκρατήσει απροσπέλαστα δεδομένα και ιστορικό αλλαγών για τις ενέργειες που έλαβαν μέρος μέσα στην βάση. Για αυτό το λόγο ήταν σε θέση να ενημερώσει έγκαιρα και έγκυρα για ύποπτες συμπεριφορές.

Κεφάλαιο 5. Συμπεράσματα

Σκοπός της παρούσας εργασίας ήταν εξ αρχής η διερεύνηση βελτίωσης της ασφαλείας που μπορεί να προσφέρει ένα σύστημα ανίχνευσης εισβολών (IDS) με την προσθήκη της τεχνολογίας του blockchain, σε μια εποχή όπου τα δεδομένα και η ασφάλεια τους είναι πιο σημαντικά από ποτέ. Μόνο μέσω αποτελεσμάτων μπορεί να δωθεί κατηγορηματική απάντηση και με γνώμονα αυτά που παρουσιάσαμε στο προηγούμενο κεφάλαιο μπορούμε πλέον να βγάλουμε τα παρακάτω συμπεράσματα.

Σε κάθε μια από τις επιθέσεις που διενεργήσαμε, ανεξαρτήτως του βαθμού τεχνικής κατάρτισης και επιπέδου δικαιωμάτων που είχε ο επιτιθέμενος, η προσθήκη του blockchain κατάφερε να μας προσφέρει την δυνατότητα διατήρησης δεδομένων και logs που περιγράφουν τις ενέργειες του δράστη. Το γεγονός αυτό είναι τεράστιας σημασίας, καθώς έχοντας απροσπέλαστα δεδομένα τα οποία τροφοδοτούν ένα IDS αυτό μπορεί όπως είδαμε να αποστείλει χρήσιμες, έγκυρες και έγκαιρες προειδοποιήσεις ή να αναχαιτίσει εξ ολοκλήρου την επίθεση. Χωρίς την ύπαρξη blockchain ένα κακόβουλο άτομο θα μπορούσε να είχε επεξεργαστεί τα δεδομένα διαγράφοντας ενδιάμεσες αλλαγές που έχουν γίνει στα logs δίχως να γίνεται αντιληπτό και συνεπώς καθιστώντας δύσκολη ακόμα και την αναγνώριση της επίθεσης. Με την ύπαρξη του blockchain όμως, έχουμε ένα σύστημα το οποίο με βεβαιότητα μας δείχνει την ροή των γεγονότων. Ουσιαστικά έχουμε ένα σύστημα καταγραφής στο οποίο μπορούμε να έχουμε μεγαλύτερο βαθμό εμπιστοσύνης από τα κλασσικά logging systems. Το αποτέλεσμα είναι πως αποκτώντας και διατηρώντας σωστά δεδομένα τα οποία τροφοδοτούν ένα IDS μειώνονται οι πιθανότητες για λάθος συναγερμούς ή αποφάσεις από αυτό. Επιπροσθέτως μια επίθεση στα logs που τροφοδοτούν το IDS, με σκοπό αυτό να μην έχει τα απαραίτητα ή τα σωστά στοιχεία για την λήψη αποφάσεων, θα γινόταν άμεσα αντιληπτή λογο της φύσης της αλυσίδας. Τέλος, επιτρέπει την καλύτερη κατανόηση των μεθόδων με τις οποίες έγινε προσπάθεια εγκληματικής ή ακόμα και λάθος ενέργειας, προσφέροντας τις απαραίτητες πληροφορίες για τη μελλοντική θωράκιση πιθανών αδύνατων σημείων.

Όταν μια τεχνολογία αποκτά δημοτικότητα, εύκολο είναι επιπόλαια να προσπαθήσουμε να την εφαρμόσουμε σε όλα μας τα προβλήματα προκειμένου την επίλυση τους, ανακαλύπτοντας αργά ή γρήγορα πως ίσως δεν αποτελεί αποδοτική λύση. Όταν αναφερόμαστε στο blockchain είναι λογικό το μυαλό μας να πηγαίνει στην μεγαλύτερη και διασημότερη υλοποίηση που υπάρχει σήμερα, είναι σημαντικό όμως να αναλογιστούμε ότι εκείνη η εφαρμογή πρόκειται και μια ομπρέλα τεχνολογιών, η βιωσιμότητα των οποίων έχει αρχίσει να ερευνάται σε άλλους τομείς – με λίγες βιώσιμες εφαρμογές. Ωστόσο οι επιμέρους τεχνολογίες που απαρτίζουν αυτό που ονομάζουμε σήμερα blockchain έχουν αποδείξει ότι μπορούν να δουλέψουν όχι μόνο σε ακαδημαϊκό επίπεδο αλλά και στον πραγματικό κόσμο. Η υλοποίηση ενός τέτοιου συστήματος όμως δεν αποτελεί πανάκεια. Ένα ακόμα σύστημα σημαίνει πιθανότατα ακόμα ένα κομμάτι τις ασφαλείας που μπορεί να αποτελέσει το ίδιο ευάλωτο σημείο. Σε μια πραγματική υλοποίηση θα πρέπει επίσης να υπολογίσουμε, πέρα από την αύξηση τις πολυπλοκότητας του συστήματος, τόσο το επιπρόσθετο χρηματικό κόστος για το ρεύμα και τα μηχανήματα (node server) που θα τρέχουν το blockchain, όσο και το υπολογιστικό κόστος. Ακόμα και με έναν πιο έξυπνο και αποδοτικό αλγόριθμο συναινέσης το κόστος σε

χρόνο και υπολογιστική ισχύ πάνω στο υπάρχων σύστημα, στη βάση δεδομένων για την συγκεκριμένη υλοποίηση, είναι μεγαλύτερη από μηδενική.

Έχοντας μελετήσει τόσο την τεχνολογία του blockchain όσο και της ανάγκης τροφοδότησης ενός συστήματος ασφαλείας με σωστά δεδομένα και έχοντας δημιουργήσει ένα απλουστευμένο IDS με τεχνολογία blockchain, άποψη του συγγραφέα είναι ότι υπάρχουν μεγάλες πιθανότητες ο συνδυασμός αυτών των δύο τεχνολογιών να μπορεί να εφαρμοστεί με επιτυχία έχοντας θετικά αποτελέσματα και στον πραγματικό κόσμο. Καθώς έχει προφανή πλεονεκτήματα η ύπαρξη πολλαπλών αντιγράφων και αντιτύπων σε πολλαπλά σημεία (μια θεμελιώδης αρχή της προστασίας και διασφάλισης των δεδομένων), όπως επίσης η ύπαρξη ενός συστήματος που αυτόβουλα μπορεί να πράξει διατηρώντας logs χωρίς να είναι αναγκαία η επιτήρηση ή η παρέμβαση από κάποιον τρίτο. Η βελτίωση τις ασφαλείας των logs υπερτερεί της αύξησης της πολυπλοκότητας του συστήματος. Η διατήρηση δεδομένων το περιεχόμενο των όποιον δεν μπορεί να αλλάξει χωρίς αυτό να γίνει άμεσα αντιληπτό βρίσκεται στο επίκεντρο της τεχνολογίας που αναλύσαμε. Για αυτό δημιουργήθηκε άλλωστε η αλυσίδα, για να είναι δυνατή η πιστοποίηση των δεδομένων που γράφονται σε αυτή, σε ένα σύστημα η όλη λειτουργία του οποίου εξαρτάται αποκλειστικά και μόνο από την ακεραιότητα των δεδομένων που λαμβάνει, η σουίτα τεχνολογιών blockchain ή ακόμα και κομμάτια αυτής μπορούν να συμβάλουν θετικά.

Βιβλιογραφία

[1] Anderson, James P. "Computer Security Threat Monitoring and Surveillance", February 1980.

[2] D. E. Denning, "An Intrusion-Detection Model," in *IEEE Transactions on Software Engineering*, vol. SE-13, no. 2, pp. 222-232, Feb. 1987, doi: 10.1109/TSE.1987.232894.

[3] “Share of households with a computer at home worldwide from 2005 to 2019” Statista, November 2021. [www.statista.com/statistics/748551/worldwide-households-with-computer].

[4] “Share of households with a computer at home in developed countries from 2005 to 2019” Statista, November 2021. [www.statista.com/statistics/748557/developed-countries-households-with-computer].

[5] “Percentage of European work force that uses a computer” Eurostat, 2020. [appsso.eurostat.ec.europa.eu/nui/show.do?dataset=isoc_ci_cm_pn2&lang=en].

[6] “Information technology (IT) spending forecast worldwide from 2012 to 2024, by segment” statista, July 2023. [www.statista.com/statistics/268938/global-it-spending-by-segment].

[7] “Cyber-attack: Europol says it was unprecedented in scale “ bbc, May 13, 2017 [www.bbc.com/news/world-europe-39907965].

[8] Reuters Staff. “Honda halts Japan car plant after WannaCry virus hits computer network”, reuters, July 21, 2017. [www.reuters.com/article/us-honda-cyberattack-idUSKBN19C0EI].

[9] “France’s Renault hit in worldwide ‘ransomware’ cyber attack” france24, May 12, 2017. [www.france24.com/en/20170512-cyberattack-ransomware-renault-worldwide-british-hospitals].

[10] “450 Computer der Bahn von "WannaCry"-Virus betroffen” spiegel, May 16, 2017. [www.spiegel.de/netzwelt/web/wannacry-450-bahn-computer-von-cyber-attacke-betroffen-a-1147921.html].

[11] Nick Statt. “Boeing production plant hit with WannaCry ransomware attack” theverge, Mar 29, 2018. [www.theverge.com/2018/3/28/17174540/boeing-wannacry-ransomware-attack-production-plant-charleston-south-carolina].

[12] “«Χάκαραν» και το ΑΠΘ στην παγκόσμια κυβερνοεπίθεση!” protothema, May 13, 2017. [www.protothema.gr/greece/article/679082/hakaran-kai-to-apth-stin-pagosmia-kuvernoepithesi].

[13] Neeta Sharma, Sneha Mary Koshy. “Monday’s Ransomware Attack Fails to Dent India, Says Minister: 10 Facts” ndtv, May 15, 2017. [www.ndtv.com/india-news/ransomware-wannacry-surfaces-in-kerala-bengal-10-facts-1693806].

[14] Heather, Landi. “Report: 40% of healthcare organizations hit by WannaCry in past 6 months” fiercehealthcare, May 29, 2019. [www.fiercehealthcare.com/tech/lingering-impacts-from-wannacry-40-healthcare-organizations-suffered-from-attack-past-6-months].

[15] McHugh, J., Christie, A., & Allen, J. “Defending Yourself: The Role of Intrusion Detection Systems” in IEEE Software 17(5), pp. October 2000, doi:10.1109/52.877859.

[16] Amir, Dahan. “Business as usual for Azure customers despite 2.4 Tbps DDoS attack” microsoft blog, October 11, 2021. [azure.microsoft.com/en-us/blog/business-as-usual-for-azure-customers-despite-24-tbps-ddos-attack].

[17] “Insider Threat Report 2018” veriato, 2018. [https://www.veriato.com/resources/whitepapers/insider-threat-report-2018].

[18] John P. Mello, Jr. “Spear phishing poses threat to industrial control systems” csoonline, September 26, 2013. [https://www.csoonline.com/article/539846/access-control-spear-phishing-poses-threat-to-industrial-control-systems.html]

[19] Debar, H., Dacier, M., & Wespi, A., “Towards a taxonomy of intrusion-detection systems”, in Computer Networks, 31(8), pp 805–822. 1999, doi:10.1016/s1389-1286(98)00017.

[20] Karen Scarfone, Peter Mell, “Guide to Intrusion Detection and Prevention Systems (IDPS)” NIST U.S Department of Commerce February 2007.

[21] Alexey Malanov, "Antivirus fundamentals: Viruses, signatures, disinfection” October 13, 2016. [https://www.kaspersky.com/blog/signature-virus-disinfection/13233].

[22] Mehra, Lekhraj & Gupta, Mukesh & BHATT, MONIKA, “An Effectual and Secure Approach for the Detection and Efficient Searching of Network Intrusion Detection System (NIDS)” in International Journal of Computer Applications, 2014 doi:10.1109/ic4.2015.7375615.

[23] Kul, G., Upadhyaya, S., & Hughes, A., “An Analysis of Complexity of Insider Attacks to Databases” in ACM Transactions on Management Information Systems, 12(1), pp 1–18, 2021, doi:10.1145/3391231.

[24] U.S. Attorney’s Office, “Fannie Mae corporate intruder sentenced to over three years in prison for attempting to wipe out Fannie Mae financial data”, FBI [https://archives.fbi.gov/archives/baltimore/press-releases/2010/ba121710.htm].

[25] David Thomas. “Pa. firm alleges ex-partners 'secretly planned' to join Armstrong Teasdale” westlaw 2/16/21. [https://today.westlaw.com/Document/I1c80a1d070ac11ebb555947e94fe83f6/View/FullText.html?transitionType=SearchItem&contextData=(sc.Default)&firstPage=true].

[26] Everton Bailey Jr. “Dallas IT worker fired for deleting city files appears to have had previous mishaps, official says” The Dallas Morning News, August 31, 2021. [https://www.dallasnews.com/news/2021/08/31/dallas-it-worker-fired-for-deleting-city-files-also-erased-data-twice-before-official-says].

[27] Kim, Zetter. ”NSA Whistleblower: The Ultimate Insider Attack” Wired, June 9, 2013.[https://www.wired.com/2013/06/nsa-leaker-ultimate-insider].

[28] Dave, Howe. “Deciphering How Edward Snowden Breached the NSA [8 Years Later]” venafi, October 14, 2022. [https://www.venafi.com/blog/deciphering-how-edward-snowden-breached-the-nsa].

[29] Stuart Haber and Scott Stornetta. “How to Time-Stamp a Digital Document” 1991 [https://link.springer.com/content/pdf/10.1007/3-540-38424-3_32.pdf].

[30] Nakamoto, S. “Bitcoin: A Peer-to-Peer Electronic Cash System” bitcoin, 2008. [https://bitcoin.org/bitcoin.pdf].

[31] “Ψηφιακό ευρώ”, europa. [https://www.ecb.europa.eu/paym/digital_euro].

[32] Veris. “Data Platform” [https://axoni.com/].

[33] El Madhoun, N., Hatin, J. and Bertin, E., “A decision tree for building IT applications: What to choose: blockchain or classical systems?”, *Annales des Telecommunications/Annals of Telecommunications*, 76(3–4), pp. 131–144. 2021, doi:10.1007/s12243-020-00814-y.

[34] Xu, Z. and Zou, C., “What can blockchain do and cannot do?”, *China Economic Journal*, 14(1), pp. 4–25. 2021, doi:10.1080/17538963.2020.1748968.

[35] Gwyneth Iredale, “Introduction to Permissioned Blockchains" 101blockchains, June 02, 2019. [https://101blockchains.com/permissioned-blockchain].

[36] Dwork, C., & Naor, M. (n.d.). “Pricing via Processing or Combatting Junk Mail” in Lecture Notes in Computer Science, pp 139–147. August 16–20, 1992, doi:10.1007/3-540-48071-4_10

[37] Adam Back, “Hashcash - A Denial of Service Counter-Measure” September 2002.

[38] Source: Hashcash documentation: hashcash.org [www.hashcash.org/docs/hashcash.html#examining_stamps].

[39] University of Cambridge, “Cambridge Bitcoin Electricity Consumption Index” https://ccaf.io/cbeci/index, accessed March 2023.

[40] U.S. Energy Information Administration - EIA - Independent Statistics and Analysis, “Electricity ” https://www.eia.gov/international/data/world/electricity/electricity-consumption?pd=2&p=0000002&u=0&f=A&v=mapbubble&a=-&i=none&vo=value&t=C&g=00000000000000000000000000000000000000000000000001&l=249-ruvvvvvfvtvnvv1vrvvvvfvvvvvvfvvvou20evvvvvvvvvvnvvvs0008&s=315532800000&e=1546300800000&, accessed February 2023.

[41] Tasca, Paolo; Tessone, Claudio J, "A Taxonomy of Blockchain Technologies: Principles of Identification and Classification" in LEDGER issue 4, 2019-02-15.

[42] King, Sunny and Scott Nadal. “PPCoin: Peer-to-Peer Crypto-Currency with Proof-of-Stake.” (2012).

[43]Chen, C., Chen, X., Yu, J., Wu, W., & Wu, D. “Impact of Temporary Fork on the Evolution of Mining Pools in Blockchain Networks: An Evolutionary Game Analysis” in IEEE Transactions on Network Science and Engineering, 8(1), 400–418. 2021, doi:10.1109/tnse.2020.3038943.

[44] El Madhoun, N., Hatin, J. and Bertin, E. “A decision tree for building IT applications: What to choose: blockchain or classical systems?” In *Annales des Telecommunications/Annals of Telecommunications*, 76(3–4), pp. 131–144. 2021, doi:10.1007/s12243-020-00814-y.

[45] ElMamy, S.B. et al. “A Survey on the Usage of Blockchain Technology for Cyber-Threats in the Context of Industry 4.0”, Sustainability, 12(21), p. 9179. 2020, doi:10.3390/su12219179.

[46] Report of the European Parliament, Parliamentary question, 2020, https://www.europarl.europa.eu/doceo/document/P-9-2020-000138_EN.html.

[47] Car-Pass, “Things you need to pay attention to” accessed 2023 [https://www.car-pass.be/en/about-car-pass].

[48] Stefanie Kowallick, “How blockchain can help to prevent odometer fraud” Bosch ConnectedWorld Blog December 5, 2017 [https://blog.bosch-si.com/blockchain/how-blockchain-can-help-to-prevent-odometer-fraud/].

[49] Archana Sristy. “Blockchain in the food supply chain - What does the future look like?” walmart, Nov. 30, 2021. [https://tech.walmart.com/content/walmart-global-tech/en_us/news/articles/blockchain-in-the-food-supply-chain.html].

[50] Source: Compiled from [https://www.cdc.gov/ecoli/2018/o157h7-04-18/index.html cdc.gov, 2018].

[51] IBM, “IBM Sterling Supply Chain Intelligence Suite: Food Trust” https://www.ibm.com/blockchain/solutions/food-trust, accessed February 2023.

[52] Putz, B., Menges, F., & Pernul, G. “A secure and auditable logging infrastructure based on a permissioned blockchain. Computers & Security”, 2019, doi:10.1016/j.cose.2019.101602.

[53] Hu, T., Xin, B., Liu, X., Chen, T., Ding, K., & Zhang, X. “Tracking the Insider Attacker: A Blockchain Traceability System for Insider Threats” in Sensors, 20(18), 5297, 2020, doi:10.3390/s20185297.

[54] Putz, B., Menges, F., & Pernul, G. “A secure and auditable logging infrastructure based on a permissioned blockchain” in Computers & Security, 101602, 2019, doi:10.1016/j.cose.2019.10160.