Ατομική Διπλωματική Εργασία

ΠΡΩΤΟΚΟΛΛΟ ΑΣΦΑΛΕΙΑΣ ΓΙΑ ΔΙΑΔΙΚΤΥΟ ΤΩΝ ΠΡΑΓΜΑΤΩΝ ΜΕ ΧΡΗΣΗ BLOCKCHAIN

Γεώργιος Χριστοφόρου

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Μάιος 2023

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΠΡΩΤΟΚΟΛΛΟ ΑΣΦΑΛΕΙΑΣ ΓΙΑ ΔΙΑΔΙΚΤΥΟ ΤΩΝ ΠΡΑΓΜΑΤΩΝ ΜΕ ΧΡΗΣΗ BLOCKCHAIN

Γεώργιος Χριστοφόρου

Επιβλέπων Καθηγητής

Βάσσος Βασιλείου

Ιάκωβος Ιωάννου

Η Ατομική Διπλωματική Εργασία υποβλήθηκε προς μερική εκπλήρωση των απαιτήσεων απόκτησης του πτυχίου Πληροφορικής του Τμήματος Πληροφορικής του Πανεπιστημίου Κύπρου

Μάιος 2023

Ευχαριστίες

Θέλω να ευχαριστήσω τους επιβλέποντες καθηγητές μου, τον καθηγητή Βάσσο Βασιλείου και τον Δρ. Ιάκωβο Ιωάννου, για την καθοδήγηση και τη βοήθεια που μου παρείχαν καθ' όλη τη διάρκεια του χρόνου που εργάζομαι για την πτυχιακή μου διατριβή. Είμαι ευγνώμων για τον χρόνο και την προσπάθεια που αφιέρωσαν υποστηρίζοντας με να την ολοκληρώσω.

Περίληψη

Στόχος της διπλωματικής μου εργασίας για την χρονιά 2022-2023, είναι η υλοποίηση ενός πρωτοκόλλου ταυτοποίηση και ασφαλής επικοινωνίας μηχανών σε ένα IoT (Internet of things) δίκτυο με την χρήση υφιστάμενων αλγορίθμων (RSA και AES) και blockchain. Συγκεκριμένα, πρόκειται για μια πειραματική εργασία στην οποία με την χρήση των πιο πάνω αλγορίθμων κρυπτογράφησης και της τεχνολογίας blockchain, προσπαθώ να ταυτοποιήσω τις διάφορες μηχανές εντός του IoT δικτύου μου και να παρέχω ασφαλή επικοινωνία μεταξύ των μηχανών αυτών. Έπειτα, αναλύω και βγάζω συμπεράσματα για τα αποτελέσματα που πήρα, σχετικά με την ασφάλεια και την επίδοση του πρωτοκόλλου μου αυτού.

Το IoT δίκτυο είναι στην ουσία 5 Raspberry Pie3 τα οποία 4 θα έχουν το ρόλο του client και ένα θα έχει το ρόλο του server(Sink node).Για την συγκεκριμένη εργασία χρησιμοποιήθηκε η τοπολογία Sink in the Middle παρόλο που το πρωτόκολλο δεν περιορίζεται από καμία τοπολογία (μπορεί να εφαρμοστεί σε όλες τις τοπολογίες η οποίες περιέχουν sink node).

Με την βοήθεια του HTTPS πρωτοκόλλου και της τεχνολογίας blockchain, επιτυγχάνεται η επικοινωνία και η ταυτοποίηση των μηχανών εντός του δικτύου. Εδώ θα γραφτούν και οι επιθέσεις που χρησιμοποιήθηκαν για επιβεβαίωση της ασφαλείας του πρωτοκόλλου καθώς και τα εργαλεία μέτρησης της επίδοσης του.

Η μεθοδολογία που χρησιμοποιήθηκε για την υλοποίηση του πρωτοκόλλου είναι η εξής:(α) Εγγραφή μηχανών στο blockchain δίκτυο. (β) Εκκίνηση των επιθέσεων στο δίκτυο. (γ) Συλλογή δεδομένων από επιθέσεις (δ) Μέτρηση επίδοσης πρωτοκόλλου και σύγκριση του με υφιστάμενα πρωτόκολλα.

Τέλος, αυτή η έρευνα υλοποιήθηκε με την χρήση της γλώσσας προγραμματισμού java.

Περιεχόμενα

Κεφάλαιο 1
Εισαγωγή

---

1. 1 Γενική εισαγωγή

Το Διαδίκτυο των πραγμάτων (IoT) είναι η διασύνδεση έξυπνων συσκευών για παρακολούθηση, ανάλυση και έλεγχο φυσικών περιβαλλόντων. Οι έξυπνες συσκευές ή smart devices μπορούν να περιλαμβάνουν τεχνολογίες όπως αισθητήρες, ενεργοποιητές και ειδικό λογισμικό, όλα προσαρμοσμένα για να εκπληρώσουν τη λειτουργικότητα του IoT. Έξυπνες πόλεις, Έξυπνα σπίτια, Το Smart Agriculture είναι μερικά παραδείγματα που χρησιμοποιούν το IoT για να επωφεληθούν από την αυτοματοποίηση καθημερινών εργασιών με σκοπό να διευκολυνθεί ο τρόπος της ζωής και γίνετε πιο αποτελεσματικοί.

Όμως ένα διαδίκτυο των πραγμάτων μπορεί να είναι ευάλωτο σε διάφορες επιθέσεις. Ένας εισβολέας μπορεί να παρακολουθεί, να μεταδίδει μηνύματα και να εισάγει ψευδείς πληροφορίες 2 στο δίκτυο. Συμβιβαστικές συσκευές κόμβου εντός των δικτύων IoT μπορεί να οδηγήσουν σε ανακριβή και / ή παραπλανητικές πληροφορίες, αποτυγχάνοντας έτσι στην επίτευξη του στόχου του δικτύου. Έτσι η προστασία κόμβων γίνεται απαραίτητη για την σωστή λειτουργεία του δικτύου.

Ένα πρωτόκολλο ασφαλείας είναι μια σειρά λειτουργιών που διασφαλίζουν την προστασία των δεδομένων. Χρησιμοποιείται με πρωτόκολλο επικοινωνίας, παρέχει ασφαλή παράδοση δεδομένων μεταξύ δύο μερών. Ο όρος αναφέρεται γενικά σε μια σειρά εξαρτημάτων που λειτουργούν παράλληλα . Για παράδειγμα, το πρότυπο 802.11i παρέχει αυτές τις λειτουργίες για ασύρματα LAN.

Στο διαδίκτυο, το TLS και το SSL χρησιμοποιούνται ευρέως για την παροχή ελέγχου ταυτότητας και κρυπτογράφησης προκειμένου να αποσταλούν αριθμοί πιστωτικών καρτών και άλλα ιδιωτικά δεδομένα σε έναν προμηθευτή. Ακολουθούν τα κύρια στοιχεία ενός πρωτοκόλλου ασφαλείας.

Έλεγχος πρόσβασης

Επαληθεύει την ταυτότητα χρήστη. Εξουσιοδοτεί την πρόσβαση σε συγκεκριμένους πόρους με βάση το επίπεδο αδειών και τις πολιτικές.

Αλγόριθμος κρυπτογράφησης

Οι κρυπτογραφικές προσεγγίσεις είναι η κρυπτογράφηση δημόσιου και ιδιωτικού κλειδιού, RSA, AES, DES. Το HTTPS και το TLS χρησιμοποιούν τις προαναφερθείσες προσεγγίσεις κρυπτογραφίας. Κάποιοι από τους αλγόριθμους αυτούς είναι το HTTPS και TLS.

Διαχείριση κλειδιών

Δημιουργήστε, διανείμετε και διατηρήστε τα κλειδιά.

Ακεραιότητα μηνύματος

Διασφαλίζει ότι το κρυπτογραφημένο μήνυμα δεν έχει παραβιαστεί.

1.2 Ορισμός και σκοπός του προβλήματος

Τα δίκτυα μεγαλώνουν και μεταβάλλονται με ολοένα και ταχύτερους ρυθμούς με σημαντικότερα τα IoT δίκτυα ιδιαίτερα για βιομηχανίες, έξυπνα νοσοκομεία και σπίτια, πράγμα που καθιστά την διαφύλαξη τους απαραίτητη. Έχοντας ως πιο κρίσιμα σημεία 1.την προστασία των δεδομένα καθώς αυτά μπορεί να είναι ζωτικής ,οικονομικής καθώς και προσωπικής σημασίας και 2.την μικρή κατανάλωση πόρων και ενέργειας λόγω της φορητότητας και του μεγέθους των μηχανών εντός του IoT δικτύου. Έχοντας αυτά υπόψη σε αυτή την πειραματική έρευνα δημιούργησα ένα πρωτόκολλο ασφαλής επικοινωνίας των μηχανών όσο αφορά τα δίκτυα IoT, έτσι σε συνδυασμό με τη φιλοσοφία του υφιστάμενου πρωτοκόλλου HTTPS, μεθόδων κρυπτογράφησης, χρονικής εγγύτητας και την τεχνολογία blockchain παρέχουν ασφάλεια από μεγάλη γκάμα επιθέσεων έχοντας παράλληλα τα κατάλληλα κριτήρια για να λειτουργούν εντός ενός IoT δικτύου.

Σε αυτό το σημείο πρέπει να αναφέρω ότι το πρωτόκολλο εφαρμόστηκε σε γλώσσα προγραμματισμού Java ενώ η φιλοσοφία του και ο τρόπος που λειτουργεί μπορεί να εφαρμοστεί στις περισσότερες νεότερες γλώσσες προγραμματισμού που έχουν τις

απαραίτητες βιβλιοθήκες κρυπτογράφησης. Αξιοσημείωτο το γεγονός, ότι στην παρούσα διπλωματική εργασία διεξάχθηκε και ένα σχετικό πείραμα για την εγκυρότητα, ακεραιότητα και των δεδομένων που μεταφέρονταν χρησιμοποιώντας το πρωτόκολλο που δημιούργησα καθώς και για την χρήση πόρων και ενέργειας που χρειάστηκαν οι μηχανές. Οι εν λόγω μηχανές είναι 5 Raspberry Pi3 τα οποία στήθηκαν σε Sink-in-the-Middle τοπολογία σε ένα τοπικό δίκτυο. Μία από αυτές έχει το ρόλο του διακομιστή που πρέπει να καταγράψει τις άλλες 4 μηχανές (που έχουν τον ρόλο του πελάτη) στο δίκτυο blockchain και έπειτα να μπορεί να τις ταυτοποίηση γρήγορα και αποδοτικά. Παράλληλα το πρωτόκολλο μας πρέπει να αντέξει τις διάφορες επιθέσεις που θα διεξάγουμε χρησιμοποιώντας το λογισμικό Kali Linux και τα διάφορα εργαλεία του.

Κεφάλαιο 2

Σχετική δουλειά και πληροφόρηση περί του αντικειμένου.

2.1 Επισκόπηση

Σε αυτό το κεφάλαιο περιγράφεται το πρωτόκολλο ασφαλείας που αναπτύχθηκε για την παρούσα διπλωματική εργασία . Θα ερευνήσουμε διάφορες προσεγγίσεις που υλοποιήθηκε κάτι παρόμοιοι με αυτό που θέλουμε στη παρούσα διπλωματική εργασία. Εννοώντας παρόμοιο δηλαδή να προσφέρει ασφαλή επικοινωνία μεταξύ IoT μηχανών και παράλληλα να χρησιμοποιεί την τεχνολογία blockchain .Επιπρόσθετα θα εξετάσουμε δούμε τι εργαλεία χρησιμοποιούν υφιστάμενα πρωτόκολλα ασφαλής επικοινωνίας για IoT συστήματα, πως μπορούμε να αξιοποιήσουμε τα εργαλεία αυτά για τις ανάγκες του δικού μας συστήματος και πως το πρωτόκολλο μας μπορεί να θεωρηθεί ασφαλής από μια γκάμα επιθέσεις.

2.2.1 Προσεγγίσεις σχετικές με την παρούσα διπλωματική εργασία

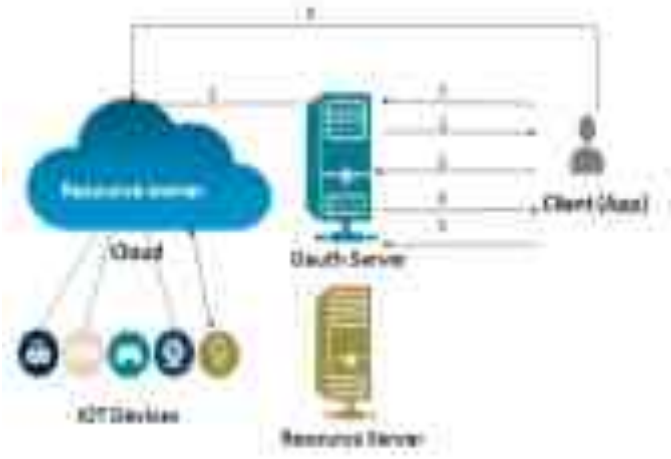
Η αντιμετώπιση επιθέσεων στο Διαδίκτυο των Πραγμάτων είναι ένα πολύ ζεστό ζήτημα στις μέρες μας. Είναι γνωστό πως εάν υπάρξει εκμετάλλευση των συσκευών που ανήκουν σε αυτόν η ακεραιότητα και η εμπιστευτικότητα της πληροφορίας δύναται να επηρεαστεί.

Λόγω του ότι αυξάνονται καθημερινά οι συσκευές που ανήκουν στο Διαδίκτυο των Πραγμάτων, πολλοί ερευνητές κλήθηκαν να αναπτύξουν και να μελετήσουν προτάσεις οι οποίες μπορούν να αντιμετωπίσουν επιθέσεις που μπορεί να εμφανιστούν σε αυτές. Συνεπώς, διάφοροι ερευνητές ανέπτυξαν πρωτόκολλα τα οποία αποσκοπούν στην διαχείριση και ταυτοποίηση μηχανών. Αυτά τα πρωτόκολλα προσπαθούν να αντιμετωπίσουν επιθέσεις που δύναται να εμφανιστούν στην επικοινωνία μεταξύ των μηχανών.

2.2.2 Πρόταση με Κεντρική μονάδα ταυτοποίησης

Στα πρωτόκολλα στα οποία πραγματοποιείται κεντρική διαχείριση των μηχανών, όλες οι μηχανές ταυτοποιούνται από την κεντρική μονάδα. Ένας από τους πιο δημοφιλείς και ευρέως χρησιμοποιούμενους ελέγχους ταυτότητας, ο μηχανισμός, ειδικά για συσκευές IoT, είναι η Ανοιχτή Εξουσιοδότηση(OAuth) [13]. Το OAuth είναι ένα ανοιχτό τυπικό πρωτόκολλο ελέγχου ταυτότητας και εξουσιοδότησης που βασίζεται σε διακριτικά που πιστοποιεί την ταυτότητα των χρηστών για πρόσβαση προστατευμένους πόρους χρησιμοποιώντας έναν αξιόπιστο κεντρικό διακομιστή.

Το Σχήμα 1 απεικονίζει τους κύριους παράγοντες που εμπλέκονται στο OAuth πρωτόκολλο εκτός από τα κύρια βήματα που πραγματοποιούνται από αυτά ηθοποιοί. Το πρωτόκολλο περιλαμβάνει τους ακόλουθους τέσσερις παράγοντες: Το ιδιοκτήτης πόρων που κατέχει τους προστατευόμενους πόρους και μπορεί παροχή πρόσβασης στους πόρους. Ο διακομιστής εξουσιοδότησης (AS) που εκδίδει διακριτικά πρόσβασης σε εξουσιοδοτημένους πελάτες. Ο πόρος διακομιστή που φιλοξενεί τους προστατευμένους πόρους και μπορεί να αποδεχτεί πρόσβαση με βάση τα εκ δοθέντα διακριτικά από το AS. Ο πελάτης που εκκινεί αίτημα πρόσβασης.



Σχήμα 1

Η επισκόπηση της διαδικασίας του πρωτοκόλλου συνοψίζεται σε έξι βήματα.

Στο Βήμα 1, ο πελάτης ζητά τη χορήγηση εξουσιοδότησης από ο κάτοχος του πόρου μέσω του διακομιστή εξουσιοδότησης.

Στο Βήμα 2, ο διακομιστής εξουσιοδότησης απαντά στον πελάτη στέλνοντάς του το όνομα χρήστη και τον κωδικό πρόσβασης.

Στο Βήμα 3, ο πελάτης ζητά ένα διακριτικό πρόσβασης από το AS με τον έλεγχο ταυτότητας του/της στο διακομιστή εξουσιοδότησης χρησιμοποιώντας το όνομα χρήστη και τον κωδικό πρόσβασης που έλαβε από το Α.Σ.

Στο Βήμα 4, το AS επαληθεύει τον χρήστη και εκδίδει ένα διακριτικό πρόσβασης στον πελάτη.

Στο Βήμα 5, ο πελάτης ζητά πρόσβαση για προστατευμένους πόρους από το AS χρησιμοποιώντας τη ληφθείσα πρόσβαση ένδειξη.

Στο Βήμα 6, το AS επαληθεύει το διακριτικό πρόσβασης με το οποίο ο πελάτης θα έχει πρόσβαση στους προστατευόμενους πόρους εάν η επαλήθευση είναι επιτυχής.

2.2.3 Προτάσεις με Αποκεντρωμένη μονάδα ταυτοποίησης

Στα πρωτόκολλα στα οποία πραγματοποιείται αποκεντρωμένη διαχείριση των μηχανών όπως το “bubbles of trust” [2]δεν υπάρχει κεντρική μονάδα ελέγχου αλλά ένα σύστημα αξιοπιστίας. Ουσιαστικά, με την προσέγγιση αυτή οι αδερφοί Hammi δημιούργησαν πολλές ασφαλής εικονικές ζώνες τις οποίες ονομάζουν φυσαλίδες αξιοπιστίας μέσα στην οποία φυσαλίδα οι χρήστες μπορούν να εμπιστεύονται ο ένας τον άλλο. Το σύστημα αυτό χρησιμοποιεί δημόσιες αλυσίδες blockchain Ethereum, για τον λόγο ότι θέλουν να είναι προσιτό από τον κόσμο. Κάθε φυσαλίδα έχει μία μηχανή ονόματι αφέντη η οποία είναι υπεύθυνη για τα μηνύματα που ανταλλάσσονται με την αλυσίδα.

Το σύστημα αυτό όπως προ είπα δημιουργήθηκε για να παρέχει ασφαλή επικοινωνία μεταξύ των χρηστών, έτσι ο Α στέλνει το μήνυμα στο blockchain, εάν το blockchain πιστοποιήσει την ταυτότητα Α, επικυρώνει τη συναλλαγή. Τέλος, ο Β μπορεί να διαβάσει το μήνυμα.

Ο αλγόριθμος που χρησιμοποίησαν είναι ο εξής:

1. Η συναλλαγή του πρώτου πελάτη αντιπροσωπεύει μια συσχέτιση εκ νέου αναζήτηση. Το απεσταλμένο μήνυμα είναι υπογεγραμμένο με το ιδιωτικό κλειδί των ακολούθων και περιέχει το εισιτήριο του ακόλουθου.

2. Όταν το blockchain λαμβάνει τη συναλλαγή επαληθεύει την ακεραιότητά του, επαληθεύοντας την υπογραφή με το δημόσιο κλειδί του ακόλουθου. Στη συνέχεια, το εισιτήριο του ακόλουθου επαληθεύεται χρησιμοποιώντας το δημόσιο κλειδί Master's, αφού αντιπροσωπεύει την οντότητα που το υπέγραψε.

3. Εάν το εισιτήριο είναι έγκυρο, τότε το blockchain αποθηκεύει έναντου grpID, του objID και του δημόσιου κλειδιού. Έτσι, αποθηκεύει(XX, YY και PubKey_F);

4. Το τέταρτο βήμα περιγράφει την περίπτωση όπου η F στέλνει ακόμη μια συναλλαγή εκτός από τη αναζήτηση του συσχετιζόμενου. Αυτή η συναλλαγή περιέχει: (1) τα δεδομένα που ανταλλάσσονται,(2) XX, (3) YY και (4) η υπογραφή ECDSA του συνόλου έθνος των προηγούμενων πεδίων χρησιμοποιώντας το ιδιωτικό κλειδί του ακόλουθου.

5. Όταν το blockchain λαμβάνει τη συναλλαγή, επαληθεύει την ακεραιότητα του επαληθεύοντας την υπογραφή με τη δημοσίευση του ακόλουθου-lic key

6. Εάν η υπογραφή είναι έγκυρη, το blockchain επαληθεύει εάν το κοινό το κλειδί που χρησιμοποιείται για την επαλήθευση της συναλλαγής αποθηκεύεται και που συσχετίζονται με το grpID και το objID που αποστέλλονται εντός της συναλλαγής.

7. Αν ο συσχετισμός είναι αποθηκευμένος και ισχύει τότε

8. Η ταυτότητα της συσκευής ελέγχεται με επιτυχία.

Μια άλλη από αυτές τις προσεγγίσεις είναι η προσέγγιση που ακολούθησε ο Abdallah Zoubir Ourad στην έρευνα του [3] για να δημιουργήσει το δικό του σύστημα ταυτοποίησης IoT μηχανών.

Σε αυτό το σύστημα χρησιμοποιείται η αλυσίδα Ethereum για να φυλάσσονται τα δεδομένα των χρηστών. Το σύστημα αυτό λειτουργεί στις εξής 3 φάσεις:

1. Έξυπνα Συμβόλαια

   Η πρώτη φάση συμβαίνει όταν ο χρήστης πιστοποιεί την ταυτότητα του έξυπνου συμβολαίου για να αποδείξει ότι είναι νόμιμος χρήστης. Ο ψευδοκώδικας παρακάτω περιγράφει τον πηγαίο κώδικα του έξυπνου συμβολαίου. Αυτή η έκδοση του έξυπνου συμβολαίου θεωρεί μόνο τον διαχειριστή ως νόμιμο χρήστη. Με άλλα λόγια, ο χρήστης που ανέπτυξε αυτό το έξυπνο συμβόλαιο στο blockchain είναι ο διαχειριστής. Η προσθήκη περισσότερων χρηστών μπορεί να επιτευχθεί με την προσθήκη μιας μεθόδου addUser() στο έξυπνο συμβόλαιο. Η διαδικάσία φαίνεται στο Σχήμα 2.

```
contract Login2
Declare Private owner, hash, token_raw, random_number
//begin constructor:
owner = msg.sender;
End Constructor

Private Function login_admin()
IF msg.sender == owner
   set random_number = random(1,100);
   set hash = keccak256(msg.sender,now,random_number);
   trigger event LoginAttempt(msg.sender, hash);
Endif
End Function
End Class
```

Σχήμα 2

Όταν οι χρήστες θέλουν να πραγματοποιήσουν έλεγχο ταυτότητας, χρησιμοποιούν τον πελάτη τους Ethereum για να καλέσουν τη μέθοδο login_admin. Η συνάρτηση login_admin δεν απαιτεί παραμέτρους και προστατεύεται από δημόσια χρήση, δηλαδή μόνο εξουσιοδοτημένοι χρήστες μπορούν να την καλέσουν επειδή ένας τροποποιητής εκτελεί την επαλήθευση της διεύθυνσης Ethereum του αποστολέα. Εάν ένας επαληθευμένος χρήστης καλέσει αυτήν τη συνάρτηση, το login_admin δημιουργεί έναν τυχαίο κατακερματισμό χρησιμοποιώντας τη

συνάρτηση rand. Στη συνέχεια, το login_admin δημιουργεί ένα διακριτικό κατακερματίζοντας τη διεύθυνση ethereum του χρήστη, τον χρόνο αποκλεισμού και τον τυχαίο κατακερματισμό που δημιουργήθηκε στο τελευταίο βήμα. Μετά από αυτό, ξεκινά ένα συμβάν για την αποστολή του διακριτικού και της διεύθυνσης του πιστοποιημένου χρήστη πίσω στη συσκευή IOT και στο χρήστη για να προχωρήσει στη φάση 2.

2. Αλληλεπίδραση χρήστη-IoT

Όταν η φάση 1 ολοκληρωθεί με επιτυχία, ο χρήστης και η συσκευή IOT λαμβάνουν ένα διακριτικό ελέγχου ταυτότητας και τη διεύθυνση Ethereum του εξουσιοδοτημένου χρήστη. Η φάση 2 συνδέει τις δύο οντότητες μεταξύ τους. Σημειώστε ότι αυτή η λύση προϋποθέτει ότι ο χρήστης γνωρίζει τη διεύθυνση - ip ή όνομα τομέα - της συσκευής IOT. Σε περίπτωση που αυτό δεν ήταν αλήθεια, η διεύθυνση της συσκευής μπορεί να σταλεί με συμβάν LoginAttempt.

Ροή υλοποίησης από την πλευρά χρήστη: Πρώτον, χρησιμοποιώντας nodejs και web3 Ethereum client, το σενάριο του χρήστη συνδέεται με το blockchain Ethereum ακούγοντας συμβάντα που προέρχονται από το αναπτυγμένο έξυπνο συμβόλαιο στη φάση 1. Όταν φτάσει το συμβάν, το σενάριο του χρήστη αποκτά πρόσβαση στον κατάλογο keystore που περιέχει τα μυστικά κλειδιά του χρήστη και εξάγει το ιδιωτικό κλειδί. Σημειώστε ότι το σενάριο χρειάζεται το πέρασμα κλειδιού για να εκτελέσει μια τέτοια ενέργεια. Στη συνέχεια, το σενάριο εξάγει το δημόσιο κλειδί από το ιδιωτικό κλειδί. Το δημόσιο κλειδί κατακερματίζεται χρησιμοποιώντας τον αλγόριθμο keccak256. Τα τελευταία 40 byte του κατακερματισμού που προκύπτει είναι η διεύθυνση Ethereum του χρήστη. Αυτή η διεύθυνση Ethereum συγκρίνεται με αυτή που ελήφθη από το συμβάν έξυπνου συμβολαίου. Αυτή είναι η επίσημη μέθοδος για να αποκτήσετε μια διεύθυνση Ethereum από έναν κατάλογο keystore. Σημειώστε ότι η χρήση του δημόσιου κλειδιού ελαχιστοποιείται και αντικαθίσταται από τη διεύθυνση Ethereum, καθώς η διεύθυνση είναι πιο σύντομη και πιο εύκολη στη χρήση. Αυτό επιτυγχάνεται χρησιμοποιώντας το keythereum για πρόσβαση στο ιδιωτικό κλειδί και το ελλειπτικό για την εξαγωγή του δημόσιου κλειδιού. Αυτές οι βιβλιοθήκες nodejs βρίσκονται στο διαδίκτυο [4, 17].

Η διαδικασία εξαγωγής της διεύθυνσης Ethereum από το κατάστημα κλειδιών ethereum του χρήστη.Αφού βεβαιωθείτε ότι η διεύθυνση Ethereum που ελήφθη από το έξυπνο συμβόλαιο είναι η διεύθυνση του χρήστη. Το σενάριο αρχίζει να δημιουργεί το μήνυμα ελέγχου ταυτότητας. Το μήνυμα μπορεί να περιγραφεί ως εξής:

(1) message=[token+src_ip+Auth_dur+PubK]

Οπου:

token: είναι το διακριτικό που λαμβάνεται από το έξυπνο συμβόλαιο.

src_ip: είναι η διεύθυνση IP από την οποία θα συνδεθεί ο χρήστης.

Auth_dur: είναι η διάρκεια της εγκυρότητας ελέγχου ταυτότητας πριν ανακληθεί και απαιτείται άλλος έλεγχος ταυτότητας.

Pubk: είναι το δημόσιο κλειδί του λογαριασμού ethereum του χρήστη.

Τέλος, το μήνυμα υπογράφεται χρησιμοποιώντας το ιδιωτικό κλειδί του λογαριασμού ethereum του χρήστη. Το ακόλουθο πακέτο ελέγχου ταυτότητας αποστέλλεται στη συσκευή IOT:

(2) μήνυμα+Υπογραφή+PubK

IOT Side Implementation Flow: Το σενάριο συσκευής IOT ξεκινά με παρόμοιο τρόπο με το σενάριο χρήστη. Συνδέεται με το έξυπνο συμβόλαιο που έχει αναπτυχθεί στη φάση 1 και ακούει το επιθυμητό συμβάν. Όταν συμβεί το συμβάν, το σενάριο λαμβάνει

το διακριτικό ελέγχου ταυτότητας και τη διεύθυνση Ethereum του επαληθευμένου χρήστη. Το σενάριο περιμένει να φτάσει το πακέτο ελέγχου ταυτότητας του χρήστη. Εάν έφτασε το πακέτο, ξεκινά η φάση επαλήθευσης. Εάν κάποιο από τα βήματα επαλήθευσης αποτύχει, το πακέτο ελέγχου ταυτότητας απορρίπτεται για να ελαχιστοποιηθεί ο φόρτος εργασίας στην ισχύ επεξεργασίας της συσκευής IOT. Η διαδικασία μετακινείται στο επόμενο βήμα μόνο εάν επαληθευτεί το τρέχον βήμα.

1. Φάση επαλήθευσης

   Βήματα φάσης επαλήθευσης:

   1.Είναι το πακέτο ελέγχου ταυτότητας και το μήνυμα στη σωστή μορφή όπως φαίνεται στις Εξισώσεις. 1 και 2?

   2.Είναι έγκυρη η υπογραφή του μηνύματος; Αυτό ελέγχεται χρησιμοποιώντας το δημόσιο κλειδί στην Εξ. 2

   3.Είναι το δημόσιο κλειδί στο πακέτο ελέγχου ταυτότητας στην Εξ. 2 παρόμοιο με αυτό στο μήνυμα στην Εξ. 1;

   4.Είναι το διακριτικό στο μήνυμα παρόμοιο με το διακριτικό από το έξυπνο συμβόλαιο;

   5.Είναι η ip προέλευσης στο μήνυμα παρόμοια με την πηγή ip του αποστολέα του πακέτου ελέγχου ταυτότητας;

   6.Κατακερματίζοντας το δημόσιο κλειδί στο μήνυμα και λαμβάνοντας τα τελευταία 40 byte ... Είναι το αποτέλεσμα παρόμοιο με τη διεύθυνση Ethereum από το έξυπνο συμβόλαιο;

   Εάν επαληθεύτηκαν όλες αυτές οι φάσεις, ο χρήστης επαληθεύεται. Διαφορετικά, το πακέτο ελέγχου ταυτότητας απορρίπτεται. Σύμφωνα με τα πρότυπα BigO, η δήλωση IF προσθέτει μια γραμμική πολυπλοκότητα εκτέλεσης στο πρόγραμμα ανάλογα με την είσοδο. Συμβολίζεται ως:

O(kn)=O(n)

Όπου k είναι σταθερά.

Στο παρακάτω Πίνακα 1 βλέπουμε αναλυτικά τις προδιαγραφές κάθε προσέγγισης.

Έρευνα	Κρυπτογράφηση	Χρονική σφραγίδα	Κατακερματισμός	Διαρρύθμιση	Αποδοτικότητα	Φορητότητα
Randa A.	ΝΑΙ	ΝΑΙ	ΝΑΙ	Κεντρική	Μέτρια	ΟΧΙ
Ourad A.	ΝΑΙ	ΝΑΙ	ΝΑΙ	Αποκεντρωμένη	Φτωχή	ΟΧΙ
Hammi M.	ΝΑΙ	ΝΑΙ	ΝΑΙ	Αποκεντρωμένη	Φτωχή	ΟΧΙ

Πίνακας 1

Από τον πιο πάνω πίνακα μπορούμε να παρατηρήσουμε ότι όλες οι προσεγγίσεις παρέχουν ασφαλή επικοινωνία μεταξύ των IoT μηχανών, παρόλα αυτά δεν είναι φορητά, δηλαδή δεν μπορεί να εφαρμοστεί σε οποιοδήποτε σύστημα χωρίς να γίνουν πρώτα οι απαραίτητες αλλαγές. Επιπλέον, η αποδοτικότητα των πιο πάνω προσεγγίσεων δεν είναι και ιδιαίτερα καλή αφού χρειάζονται εξιδεικευμένα προγράμματα να τρέχουν στο παρασκήνιο.

2.3 Έρευνα Υποβάθρου

Ένα πρωτόκολλο επικοινωνίας καθορίζει τους κανόνες και τις συμφωνίες για τις αλληλεπιδράσεις των συνεργαζόμενων κατανεμημένων διαδικασιών. Σε ένα πρωτόκολλο ορίζονται τα γεγονότα πρωτοκόλλου με τις σχέσεις τους. Η καθολικότητα μιας τέτοιας προδιαγραφής περιορίζεται στο ότι ορισμένες από τις χρονικές σχέσεις μεταξύ γεγονότων δεν είναι χαρακτηριστικά των ίδιων των γεγονότων αλλά εξαρτώνται από τη θέση στην οποία βασίζεται η προδιαγραφή.

Έχουν γίνει πολλές μελέτες στη βιβλιογραφία σχετικά με τη σύγκριση απόδοσης των πρωτοκόλλων επικοινωνίας που χρησιμοποιούνται στο IoT. Αυτές οι μελέτες χρησιμοποιούν διαφορετικές πλατφόρμες υλικού, όπως arduino, raspberry pi, κ.λπ. και τεχνολογίες όπως ενσύρματο, ασύρματο κυψελοειδές και μη κυψελοειδές κ.λπ. Ως αποτέλεσμα, δεν υπάρχει γενικό μοντέλο, προσέγγιση ή σημείο αναφοράς για σύγκριση απόδοσης, επειδή τα εύρη οι εφαρμογές που βασίζονται στο IoT είναι τόσο ευέλικτες. Στην παρούσα διπλωματική εργασία θα ασχοληθούμε με τις μελέτες που χρησιμοποίησαν Raspberry Pi. Επομένως, αναλύουμε πρωτόκολλα επικοινωνίας IoT και παρουσιάζουμε μια σύγκριση σύμφωνα με την τεχνολογία IoT, τη μέτρηση απόδοσης, την πειραματική αξιολόγηση απόδοσης και το περιβάλλον προσομοίωσης στο πιο κάτω πίνακα.

2.3.1 Οι Chaudhary et al. [4]εξέτασε τρία πρωτόκολλα επιπέδου εφαρμογής IoT, συμπεριλαμβανομένων των MQTT, CoAP και AMQP. Επίσης, πραγματοποίησαν πειράματα σε πρωτόκολλα MQTT, CoAP και AMQP σε ενσύρματες, ασύρματες και 4G συνδέσεις. Χρησιμοποίησαν το Raspberry Pi3 ως συσκευές IoT.

2.3.2 Ο Sarafov [5] πραγματοποίησε αναλύσεις απόδοσης σε WebSocket, MQTT και CoAP. Τα πειράματα διεξήχθησαν μέσω τοπικού WiFi και χρησιμοποιήθηκε ένα Rasperry Pi B στην πλευρά του πελάτη και ένας φορητός υπολογιστής με επεξεργαστή i7 στην πλευρά του διακομιστή. Κατά την ανάλυση των αποτελεσμάτων, χρησιμοποιήθηκε μια μαθηματική μέθοδος για τη σύγκριση των πρωτοκόλλων σύμφωνα με τις παραμέτρους διεκπεραίωσης σε διαφορετικούς ρυθμούς απώλειας πακέτων και εξετάστηκε η επίδραση των μεγεθών πλαισίων.

2.3.3 Οι Tandale et al. [6] πραγματοποίησε μια εμπειρική μελέτη για τα CoAP, MQTT και Rest. Το CoAP παρουσίασε την καλύτερη λύση για μικρά ωφέλιμα φορτία σύμφωνα με πειραματικά αποτελέσματα.

2.3.4 Ο Nastase [7] πραγματοποίησε μια σύγκριση των CoAP, MQTT, XMPP, AMQP, HTTP και WebSocket αντιμετωπίζοντας διαφορετικές προοπτικές όπως ο χρόνος μετ' επιστροφής, το μέσο μέγεθος πακέτου κ.λπ. Η πειραματική ρύθμιση διεξήχθη με προσομοίωση ενός δικτύου IoT στο Raspberry Pi 3 .

2.3.5 Οι Chen et al. [8] συνέκρινε πρωτόκολλα MQTT, CoAP, DDS και προσαρμοσμένα UDP για ιατρικές εφαρμογές κάτω από ένα δίκτυο ασύρματης

πρόσβασης με περιορισμένη πρόσβαση. Δημιουργήθηκε μια υποδομή υλικού με ιατρικούς αισθητήρες και λογισμικό για την προσομοίωση των υπαρχουσών υλοποιήσεων. Για τη μέτρηση και την ανάλυση της απόδοσης των πρωτοκόλλων χρησιμοποιήθηκαν μετρήσεις απόδοσης δικτύου, όπως κατανάλωση εύρους ζώνης, καθυστέρηση, απώλεια πακέτων και γενικά έξοδα ελέγχου.

2.3.6 Οι Hedi et al. [9] συνέκρινε το MQTT και το CoAP, λαμβάνοντας υπόψη τον χρόνο ανταλλαγής δεδομένων σε διαφορετικά σενάρια.

2.3.7 Οι hota και Kim [10] πραγματοποίησαν τη σύγκριση μεταξύ MQTT και CoAP εφαρμόζοντας ένα σενάριο IoT στο ESP8266. Οι Al-Masri et al. [11]τόνισε τις διαφορετικές μεθόδους των πρωτοκόλλων και την εφαρμογή τους σε διάφορες ρυθμίσεις IoT και τα προβλήματα, τα δυνατά σημεία και τους περιορισμούς.

2.3.8 Σχολιασμός προ υπάρχοντών πρωτοκόλλων ασφαλής επικοινωνίας IoT μηχανών.

Όπως μπορούμε να δούμε στο πιο κάτω Πίνακα 2 όλα τα πρωτόκολλα που υπάρχουν μέχρι σήμερα είναι απαραίτητο να έχουν ένα είδος κρυπτογράφησής, κατακερματισμού και χρονικής σφραγίδας για να μπορούν να προστατευτούν από διάφορες επιθέσεις τις οποίες θα δούμε αναλυτικά στο Κεφάλαιο 3.

Article	Analyzed IoT Communication Protocol	IoT Technology	Performance Metrics	Experimental Performance Evaluation	Simulation Environment	Encryption RSA/AES	Hash	Timestamp
Chaudhary,2017	CoAP, MQTT, AMQP	Wired, Wi-Fi, 3.5/4G Raspberry Pi3	Number Of Transmitted Message, Bandwidth Usage	Yes	Wireshark, RabbitMQ Eclipse Mosquitto, Libcoap Server, Python	Both	Both	Yes
Sanchez, 2017	WebSocket, MQTT, CoAP	Raspberry Pi B, Wi-Fi	Packet Loss Rates, Throughput	Yes	Wireshark, Linux Netem	Both	Both	Yes
Tandale, 2017	MQTT, CoAP, Rest	Raspberry Pi3, 4G, High speed broadband connection	Upload-Download Time, Bandwidth Consumption,	Yes	Mosquitto, Anconap, Django	Both	Both	Yes
Naman, 2017	CoAP, MQTT, XMPP, AMQP, HTTP, WebSocket	Raspberry Pi3, 802.11n Wireless	Round-Trip Time, Average Of Packet Size, Useful Packet Ratio,	Yes	Python	Both	Both	Yes
Chen, 2016	MQTT, CoAP, DDS, Custom UDP	Raspberry Pi2, Arduino Uno	Bandwidth Consumption, Latency, Packet Loss, Control Overhead	Yes	Netem, Token Bucket Filter, Wireshark	Both	Both	Yes
Hesk, 2017	MQTT, CoAP	Raspberry Pi3	Data exchange time	Yes	Paho Python Client, WebIOPi	Both	Both	Yes
Thota, 2018	MQTT, CoAP	Raspberry Pi, ESP8266	Packet Loss Rate	Yes	Python toThingz	Both	Both	Yes
Al-Masri, 2020	HTTP, MQTT, AMQP, CoAP, XMPP, DDS	Raspberry Pi3	Latency, Throughput, Memory and Power consumption	Yes	Mosquitto, Wireshark, Berry Wise MQTT broker	Both	Both	Yes

Πίνακας 2

2.4 Προσεγγίσεις Ασφαλείας

2.4.1 MQTT

Το MQTT είναι ένα τυπικό πρωτόκολλο ανταλλαγής μηνυμάτων OASIS για το Internet of Things (IoT). Έχει σχεδιαστεί ως ένα εξαιρετικά ελαφρύ μέσο μεταφοράς μηνυμάτων δημοσίευσης/συνδρομής που είναι ιδανικό για τη σύνδεση απομακρυσμένων συσκευών με μικρό αποτύπωμα κώδικα και ελάχιστο εύρος ζώνης δικτύου. Το MQTT χρησιμοποιείται σήμερα σε μια μεγάλη ποικιλία βιομηχανιών, όπως η αυτοκινητοβιομηχανία, η μεταποίηση, οι τηλεπικοινωνίες, το πετρέλαιο και το φυσικό αέριο κ.λπ.

2.4.2 CoAP

Το Πρωτόκολλο Περιορισμένης Εφαρμογής (CoAP) είναι ένα εξειδικευμένο πρωτόκολλο μεταφοράς ιστού για χρήση με περιορισμένους κόμβους και περιορισμένα δίκτυα στο Internet of Things. Το CoAP έχει σχεδιαστεί για να επιτρέπει σε απλές, περιορισμένες συσκευές να συνδέονται στο IoT ακόμη και μέσω περιορισμένων δικτύων με χαμηλό εύρος ζώνης και χαμηλή διαθεσιμότητα. Χρησιμοποιείται γενικά για εφαρμογές μηχανής με μηχανή (M2M), όπως η έξυπνη ενέργεια και ο αυτοματισμός κτιρίων. Το πρωτόκολλο σχεδιάστηκε από την Ομάδα Εργασίας Μηχανικής Διαδικτύου (IETF), το CoAP καθορίζεται στο IETF RFC 7252.

2.4.3 AMQP

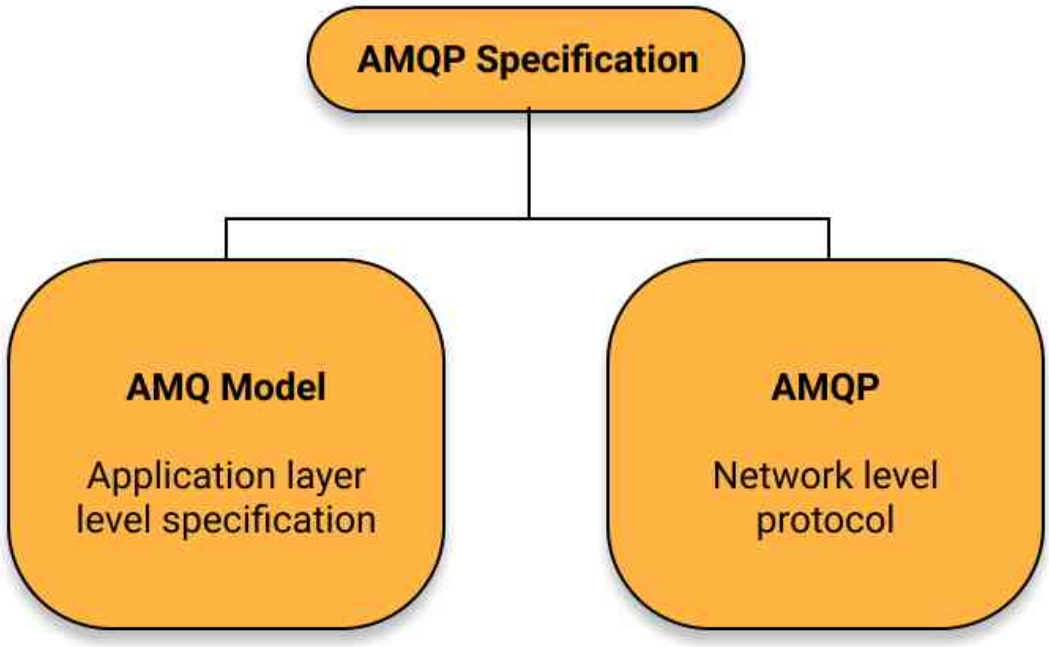
Το Advanced Message Queuing Protocol (AMQP) δημιουργείται ως ένα ανοιχτό πρότυπο πρωτόκολλο που επιτρέπει τη λειτουργικότητα των μηνυμάτων μεταξύ συστημάτων, ανεξάρτητα από τον προμηθευτή ή την πλατφόρμα που χρησιμοποιείται. Με το AMQP, μπορείτε να χρησιμοποιήσετε όποια βιβλιοθήκη πελατών συμβατή με AMQP θέλετε και οποιονδήποτε μεσίτη συμβατό με AMQP θέλετε. Οι πελάτες μηνυμάτων που χρησιμοποιούν AMQP είναι εντελώς αγνωστικιστές.

Το AMQP είναι ένα πρωτόκολλο επιπέδου εφαρμογής που επιτρέπει στις εφαρμογές-πελάτες να συνομιλούν με τον διακομιστή και να αλληλοεπιδρούν. Ωστόσο, το AMQP δεν πρέπει να θεωρείται απλώς ένα πρωτόκολλο που χρησιμοποιείται για επικοινωνία μέσω καλωδίου. Το AMQP ορίζει τόσο το πρωτόκολλο του επιπέδου δικτύου όσο και μια αρχιτεκτονική υψηλού επιπέδου για τους μεσίτες μηνυμάτων.

Ορίζει ένα σύνολο δυνατοτήτων μηνυμάτων που πρέπει να διατίθενται από μια εφαρμογή διακομιστή συμβατή με AMQP (όπως το RabbitMQ).

Συμπεριλαμβανομένων κανόνων για το πώς τα μηνύματα πρέπει να δρομολογούνται και να αποθηκεύονται στον μεσίτη για να ακολουθήσουν το μοντέλο AMQ.

Το RabbitMQ είναι ένα σύστημα ανταλλαγής μηνυμάτων που χρησιμοποιεί το AMQP 0.9.1 ως βάση για ένα σύνολο προτύπων που ελέγχουν ολόκληρη τη διαδικασία μετάδοσης μηνυμάτων. Προσδιορισμός του πρωτοκόλλου αυτού φαίνεται στο παρακάτω Σχήμα 3.



Σχήμα 3

2.4.4 XMPP

Το XMPP είναι το επεκτάσιμο πρωτόκολλο ανταλλαγής μηνυμάτων και παρουσίας, ένα σύνολο ανοιχτών τεχνολογιών για ανταλλαγή άμεσων μηνυμάτων, παρουσία, συνομιλία πολλαπλών μερών, φωνητικές κλήσεις και βιντεοκλήσεις, συνεργασία, ελαφρύ ενδιάμεσο λογισμικό, διανομή περιεχομένου και γενικευμένη δρομολόγηση δεδομένων XML.

Το XMPP αναπτύχθηκε αρχικά στην κοινότητα ανοιχτού κώδικα Jabber για να παρέχει μια ανοιχτή, αποκεντρωμένη εναλλακτική λύση στις κλειστές υπηρεσίες άμεσων μηνυμάτων εκείνη την εποχή. Το XMPP προσφέρει πολλά βασικά πλεονεκτήματα σε σχέση με τέτοιες υπηρεσίες:

Ανοιχτό — τα πρωτόκολλα XMPP είναι δωρεάν, ανοιχτά, δημόσια και εύκολα κατανοητά. Επιπλέον, υπάρχουν πολλαπλές υλοποιήσεις με τη μορφή πελατών, διακομιστών, στοιχείων διακομιστή και βιβλιοθηκών κώδικα.

Πρότυπο — η Ομάδα Εργασίας Μηχανικής Διαδικτύου (IETF) έχει επισημοποιήσει τα βασικά πρωτόκολλα ροής XML ως εγκεκριμένη τεχνολογία ανταλλαγής άμεσων μηνυμάτων και παρουσίας. Οι προδιαγραφές XMPP δημοσιεύθηκαν ως RFC 3920 και RFC 3921 το 2004 και το XMPP Standards Foundation συνεχίζει να δημοσιεύει πολλά πρωτόκολλα επέκτασης XMPP. Το 2011 τα βασικά RFC αναθεωρήθηκαν, με αποτέλεσμα τις πιο ενημερωμένες προδιαγραφές (RFC 6120, RFC 6121 και RFC 7622).

Αποδεδειγμένα — οι πρώτες τεχνολογίες Jabber/XMPP αναπτύχθηκαν από τον Jeremie Miller το 1998 και τώρα είναι αρκετά σταθερές. εκατοντάδες προγραμματιστές εργάζονται σε αυτές τις τεχνολογίες, υπάρχουν δεκάδες χιλιάδες διακομιστές XMPP που εκτελούνται σήμερα στο Διαδίκτυο και εκατομμύρια άνθρωποι χρησιμοποιούν το XMPP για ανταλλαγή άμεσων μηνυμάτων μέσω δημόσιων υπηρεσιών όπως το Google Talk και η ανάπτυξη XMPP σε οργανισμούς παγκοσμίως.

Αποκεντρωμένη — η αρχιτεκτονική του δικτύου XMPP είναι παρόμοια με το ηλεκτρονικό ταχυδρομείο. Ως αποτέλεσμα, ο καθένας μπορεί να τρέξει τον δικό του διακομιστή XMPP, επιτρέποντας σε άτομα και οργανισμούς να αναλάβουν τον έλεγχο της επικοινωνιακής τους εμπειρίας.

Ασφαλής — οποιοσδήποτε διακομιστής XMPP μπορεί να είναι απομονωμένος από το δημόσιο δίκτυο (π.χ. σε εταιρικό intranet) και στις βασικές προδιαγραφές XMPP έχει ενσωματωθεί ισχυρή ασφάλεια με χρήση SASL και TLS. Επιπλέον, η κοινότητα προγραμματιστών XMPP εργάζεται ενεργά στην κρυπτογράφηση από άκρο σε άκρο για να ανεβάσει ακόμη περισσότερο τη γραμμή ασφαλείας.

Επεκτάσιμο — χρησιμοποιώντας τη δύναμη της XML, ο καθένας μπορεί να δημιουργήσει προσαρμοσμένη λειτουργικότητα πάνω από τα βασικά πρωτόκολλα. Για τη διατήρηση της λειτουργικότητας, κοινές επεκτάσεις δημοσιεύονται στη σειρά XEP, αλλά δεν απαιτείται τέτοια δημοσίευση και οι οργανισμοί μπορούν να διατηρήσουν τις δικές τους ιδιωτικές επεκτάσεις εάν το επιθυμούν.

Ευέλικτο — Οι εφαρμογές XMPP πέρα από το IM περιλαμβάνουν διαχείριση δικτύου, διανομή περιεχομένου, εργαλεία συνεργασίας, κοινή χρήση αρχείων, gaming, απομακρυσμένη παρακολούθηση συστημάτων, υπηρεσίες web, ελαφρύ ενδιάμεσο λογισμικό, cloud computing και πολλά άλλα.

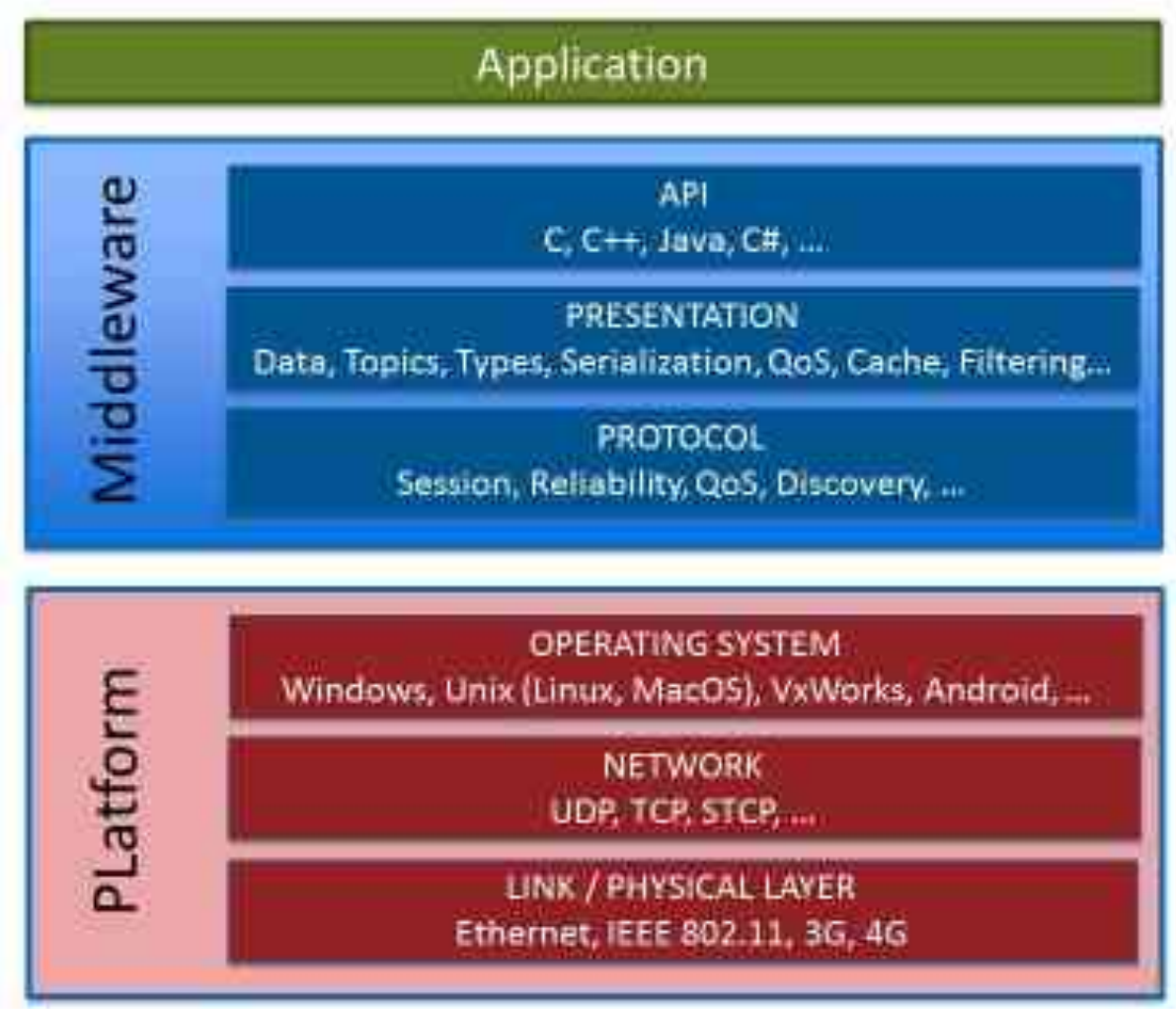
Διαφορετικές — ένα ευρύ φάσμα εταιρειών και έργων ανοιχτού κώδικα χρησιμοποιούν XMPP για τη δημιουργία και την ανάπτυξη εφαρμογών και υπηρεσιών σε πραγματικό χρόνο. ποτέ δεν θα «κλειδωθείτε» όταν χρησιμοποιείτε τεχνολογίες XMPP.

2.4.5 DDS

Η OMG Data Distribution Service (DDS™) είναι ένα πρωτόκολλο ενδιάμεσου λογισμικού και ένα πρότυπο API για συνδεσιμότητα με επίκεντρο τα δεδομένα από το Object Management Group® (OMG®). Ενσωματώνει τα στοιχεία ενός συστήματος μαζί, παρέχοντας συνδεσιμότητα δεδομένων χαμηλής καθυστέρησης, εξαιρετική αξιοπιστία και μια επεκτάσιμη αρχιτεκτονική που χρειάζονται οι επιχειρήσεις και οι κρίσιμες για την αποστολή εφαρμογές Internet of Things (IoT).

Σε ένα κατανεμημένο σύστημα, το ενδιάμεσο λογισμικό είναι το επίπεδο λογισμικού που βρίσκεται μεταξύ του λειτουργικού συστήματος και των εφαρμογών αυτό

αναπαρίσταται στο πιο κάτω Σχήμα 4. Επιτρέπει στα διάφορα στοιχεία ενός συστήματος να επικοινωνούν πιο εύκολα και να μοιράζονται δεδομένα. Απλοποιεί την ανάπτυξη κατανεμημένων συστημάτων επιτρέποντας στους προγραμματιστές λογισμικού να εστιάζουν στον συγκεκριμένο σκοπό των εφαρμογών τους και όχι στον μηχανισμό της μετάδοσης πληροφοριών μεταξύ εφαρμογών και συστημάτων.



Σχήμα 4

2.4.6 WebSocket

Το WebSocket μάς επιτρέπει να δημιουργούμε εφαρμογές «σε πραγματικό χρόνο» που είναι πιο γρήγορες και απαιτούν λιγότερα έξοδα από τα παραδοσιακά πρωτόκολλα API. Μερικές φορές αναφέρεται ως πρωτόκολλο επικοινωνίας υπολογιστή προηγμένης τεχνολογίας, το WebSocket απαιτείται για τη δημιουργία του καναλιού επικοινωνίας

πελάτη-διακομιστή. Σε αυτήν την ανάρτηση, θα εξετάσουμε τι είναι το WebSocket, πώς λειτουργεί και ορισμένα από τα σχετικά ζητήματα ασφάλειας που πρέπει να αντιμετωπιστούν.

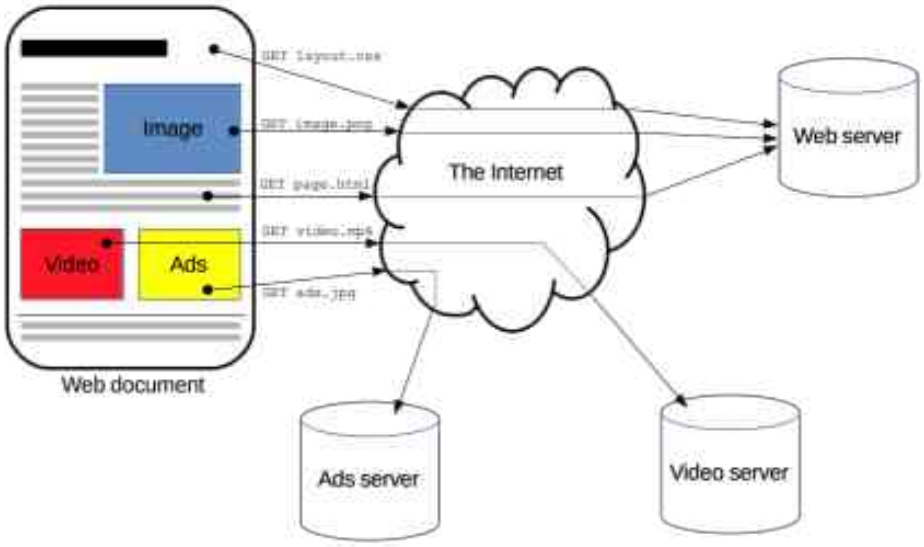
Σύμφωνα με τον συμβατικό ορισμό, το WebSocket είναι ένα πρωτόκολλο διπλής όψης που χρησιμοποιείται κυρίως στο κανάλι επικοινωνίας πελάτη-διακομιστή. Είναι αμφίδρομης φύσης που σημαίνει ότι η επικοινωνία συμβαίνει πέρα δώθε μεταξύ πελάτη-διακομιστή.

Η σύνδεση, που αναπτύχθηκε με τη χρήση του WebSocket, διαρκεί για όσο χρονικό διάστημα κάποιο από τα συμμετέχοντα μέρη την σταματήσει. Μόλις ένα μέρος διακόψει τη σύνδεση, το δεύτερο μέρος δεν θα μπορεί να επικοινωνήσει καθώς η σύνδεση διακόπτεται αυτόματα στο μπροστινό μέρος του.

Το WebSocket χρειάζεται υποστήριξη από το HTTP για να ξεκινήσει τη σύνδεση. Μιλώντας για τη χρησιμότητά του, είναι η ραχοκοκαλιά για την ανάπτυξη σύγχρονων διαδικτυακών εφαρμογών όταν πρόκειται για απρόσκοπτη ροή δεδομένων και ποικίλη μη συγχρονισμένη κίνηση.

2.4.7 HTTP

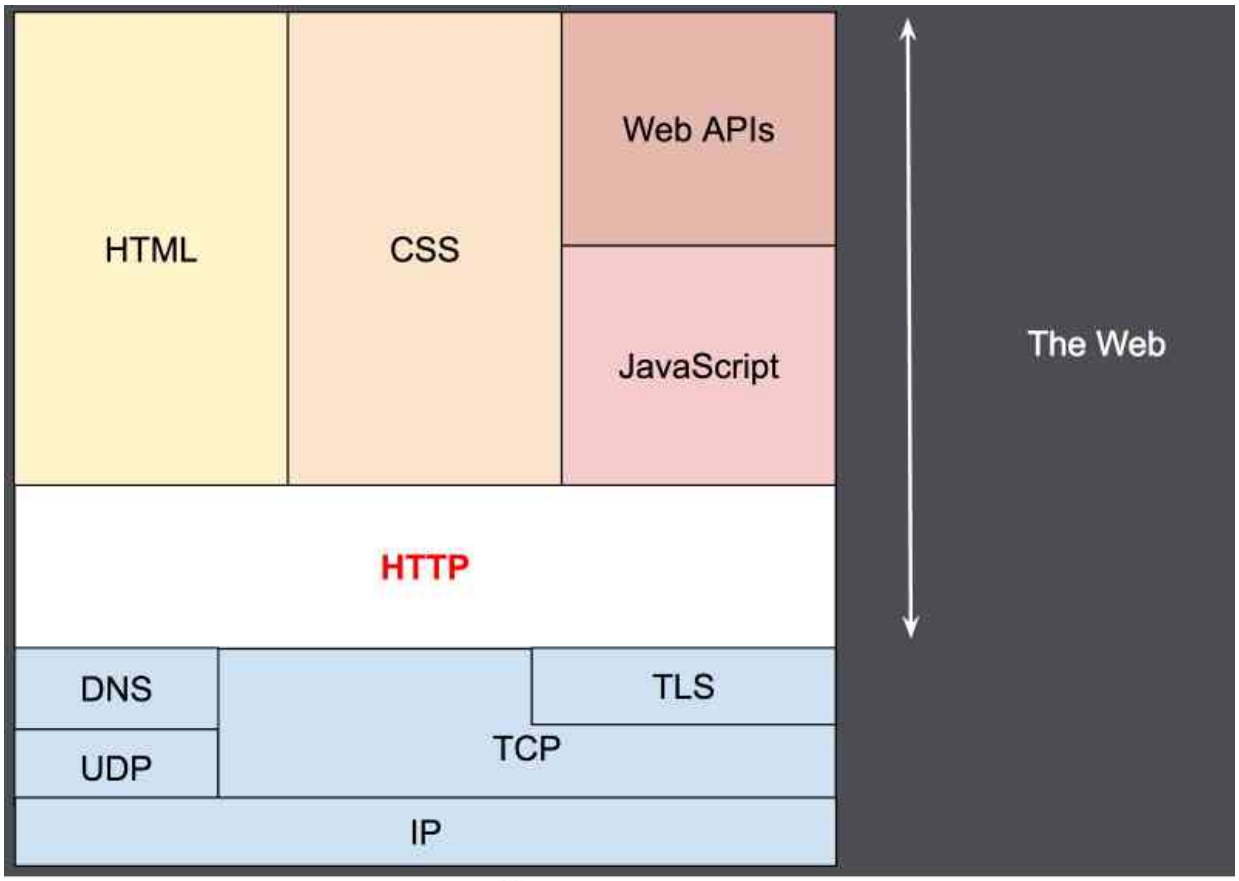
Το HTTP είναι ένα πρωτόκολλο για την ανάκτηση πόρων όπως έγγραφα HTML. Είναι το θεμέλιο οποιασδήποτε ανταλλαγής δεδομένων στον Ιστό και είναι ένα πρωτόκολλο πελάτη-διακομιστή, που σημαίνει ότι τα αιτήματα ξεκινούν από τον παραλήπτη, συνήθως το πρόγραμμα περιήγησης Ιστού. Ένα πλήρες έγγραφο ανακατασκευάζεται από τα διάφορα υπό έγγραφα που λαμβάνονται, για παράδειγμα, κείμενο, περιγραφή διάταξης, εικόνες, βίντεο, σενάρια και άλλα, το οποίο αναπαρίσταται στο πιο κάτω Σχήμα 5.



Σχήμα 5

Ένα έγγραφο Ιστού είναι η σύνθεση διαφορετικών πόρων, το οποίο αναπαρίσταται στο πιο κάτω Σχήμα 6.

Οι πελάτες και οι διακομιστές επικοινωνούν ανταλλάσσοντας μεμονωμένα μηνύματα (σε αντίθεση με μια ροή δεδομένων). Τα μηνύματα που αποστέλλονται από τον πελάτη, συνήθως ένα πρόγραμμα περιήγησης Ιστού, ονομάζονται αιτήματα και τα μηνύματα που αποστέλλονται από τον διακομιστή ως απάντηση ονομάζονται απαντήσεις.



Σχήμα 6

Το HTTP ως πρωτόκολλο επιπέδου εφαρμογής, πάνω από το TCP (επίπεδο μεταφοράς) και το IP (επίπεδο δικτύου) και κάτω από το επίπεδο παρουσίασης. Σχεδιασμένο στις αρχές της δεκαετίας του 1990, το HTTP είναι ένα επεκτάσιμο πρωτόκολλο που έχει εξελιχθεί με την πάροδο του χρόνου. Είναι ένα πρωτόκολλο επιπέδου εφαρμογής που αποστέλλεται μέσω TCP ή μέσω μιας κρυπτογραφημένης με TLS σύνδεση TCP, αν και θεωρητικά θα μπορούσε να χρησιμοποιηθεί οποιοδήποτε αξιόπιστο πρωτόκολλο μεταφοράς. Λόγω της επεκτασιμότητας του, χρησιμοποιείται όχι μόνο για τη λήψη εγγράφων υπερκειμένου, αλλά και εικόνων και βίντεο ή για τη δημοσίευση περιεχομένου σε διακομιστές, όπως με τα αποτελέσματα φόρμας HTML. Το HTTP μπορεί επίσης να χρησιμοποιηθεί για την ανάκτηση τμημάτων εγγράφων για ενημέρωση ιστοσελίδων κατά παραγγελία.

2.5 Προσεγγίσεις Κρυπτογράφησης

2.5.1 RSA

Ο αλγόριθμος RSA είναι ασύμμετρος αλγόριθμος κρυπτογράφησης. Ασυμμετρικός σημαίνει πως δουλεύει με δύο διαφορετικά κλειδιά π.χ. ένα Public και ένα Private κλειδί. Όπως μπορεί να διαπιστωθεί και από το όνομα το Public κλειδί δίνεται σε όλους ενώ το Private κλειδί παραμένει κρυφό.

Παράδειγμα ασύμμετρης κρυπτογράφησης:

1. Ο client στέλνει το Public κλειδί του στων Server καθώς του ζητάει και κάποια δεδομένα.
2. Ο Server κρυπτογραφεί α δεδομένα με το Public κλειδί του Client και του τα στέλνει.
3. Ο Client παραλαμβάνει τα κρυπτογραφημένα δεδομένα και τα αποκρυπτογραφεί χρησιμοποιώντας το Private κλειδί του.

Αφού είναι ασύμμετρικός αλγόριθμος, κανένας εκτός από τον Client δεν μπορεί να αποκρυπτογραφήσει τα δεδομένα ακόμα και αν έχει το Public κλειδί του Client.

Η ιδέα του RSA βασίζεται στο γεγονός ότι είναι εξαιρετικά δύσκολο να παραγοντοποιήσεις ένα μεγάλο ακέραιο αριθμό. Το Public κλειδί αποτελείται από 2 αριθμούς όπου είναι ο πολλαπλασιασμός των δύο μεγάλων πρώτων ακεραίων. Το Private κλειδί προέρχεται επίσης από τους ίδιους δύο πρώτους αριθμούς. Έτσι, εάν κάποιος μπορεί να παραγοντοποιήσει τον μεγάλο αριθμό, το Private κλειδί παραβιάζεται. Επομένως, η ισχύς της κρυπτογράφησης εξαρτάται αποκλειστικά από το μέγεθος του κλειδιού, η ισχύς της κρυπτογράφησης αυξάνεται εκθετικά. Τα κλειδιά RSA μπορεί να έχουν μήκος 1024 ή 2048 bit, αλλά οι ειδικοί πιστεύουν ότι τα κλειδιά 1024 bit θα μπορούσαν να σπάσουν στο εγγύς μέλλον έτσι χρησιμοποιούμε 2048 bit κλειδιά. Αλλά μέχρι τώρα φαίνεται να είναι ανέφικτο έργο.

2.5.2 AES

Advanced Encryption Standard (AES) είναι μια προδιαγραφή για την κρυπτογράφηση των ηλεκτρονικών δεδομένων που καθιερώθηκε από το Εθνικό Ινστιτούτο Προτύπων και Τεχνολογία των ΗΠΑ (NIST) το 2001. Ο AES χρησιμοποιείται ευρέως σήμερα,

καθώς είναι πολύ ισχυρότερος από τον DES και τον τριπλό DES, παρόλο που είναι πιο δύσκολο να εφαρμοστεί.

- Είναι μπλοκ κρυπτογράφησης, δηλαδή «σπάζει» τα δεδομένα σε μπλοκ και τα κρυπτογραφεί.
- Το μέγεθος του κλειδιού μπορεί να είναι 128/192/256 bits.
- Κρυπτογραφεί τα δεδομένα σε μπλοκ των 128 bits το καθένα.

Αυτό σημαίνει πως παίρνει 128 bits δεδομένων και δίνει 128 bits κρυπτογραφημένων δεδομένων. Ο AES βασίζεται στην αρχή δικτύου αντικατάστασης-μετάθεσης, που σημαίνει ότι εκτελείται χρησιμοποιώντας μια σειρά συνδεδεμένων λειτουργιών που περιλαμβάνει την αντικατάσταση και το ανακάτεμα των δεδομένων εισόδου. Το σετ εντολών AES είναι πλέον ενσωματωμένο στην CPU (προσφέρει απόδοση πολλών GB/s) για τη βελτίωση της ταχύτητας και της ασφάλειας των εφαρμογών που χρησιμοποιούν το AES για κρυπτογράφηση και αποκρυπτογράφηση. Παρόλο που έχουν περάσει 20 χρόνια από την εισαγωγή του, δεν καταφέραμε να σπάσουμε τον αλγόριθμο AES καθώς είναι ανέφικτος ακόμη και με την τρέχουσα τεχνολογία.

2.6 Προσεγγίσεις Κατακερματισμού

SHA-256:

Το SHA-256 σημαίνει Secure Hash Algorithm 256-bit και χρησιμοποιείται για κρυπτογραφική ασφάλεια.

Οι κρυπτογραφικοί αλγόριθμοι κατακερματισμού παράγουν μη αναστρέψιμους και μοναδικούς κατακερματισμούς. Όσο μεγαλύτερος είναι ο αριθμός των πιθανών κατακερματισμών, τόσο μικρότερη είναι η πιθανότητα δύο τιμές να δημιουργήσουν τον ίδιο κατακερματισμό.

2.7 Ασφαλή Πρωτόκολλα Επικοινωνίας

2.7.1 HTTPS

Το πρωτόκολλο ασφαλούς μεταφοράς υπερκειμένου είναι η ασφαλής έκδοση του HTTP, το οποίο είναι το κύριο πρωτόκολλο που χρησιμοποιείται για την αποστολή δεδομένων μεταξύ ενός προγράμματος περιήγησης Ιστού και ενός ισοτόπου. Το HTTPS είναι κρυπτογραφημένο προκείμενο να αυξηθεί η ασφάλεια της μεταφοράς δεδομένων. Αυτό είναι ιδιαίτερα σημαντικό όταν οι χρήστες μεταδίδουν ευαίσθητα δεδομένα, όπως μέσω σύνδεσης σε τραπεζικό λογαριασμό, υπηρεσία email ή πάροχο ασφάλισης υγείας.

Οποιοσδήποτε ιστότοπος, ειδικά εκείνοι που απαιτούν διαπιστευτήρια σύνδεσης, θα πρέπει να χρησιμοποιούν HTTPS. Στα σύγχρονα προγράμματα περιήγησης ιστού όπως το Chrome, οι ιστότοποι που δεν χρησιμοποιούν HTTPS επισημαίνονται διαφορετικά από αυτούς που χρησιμοποιούν.

Το HTTPS χρησιμοποιεί ένα πρωτόκολλο κρυπτογράφησης για την κρυπτογράφηση των επικοινωνιών. Το πρωτόκολλο ονομάζεται Transport Layer Protocol (TLS), αν και παλαιότερα ήταν γνωστό ως Secure Sockets Layer (SSL). Αυτό το πρωτόκολλο ασφαλίζει τις επικοινωνίες χρησιμοποιώντας αυτό που είναι γνωστό ως υποδομή ασύμμετρου δημοσίου κλειδιού. Αυτός ο τύπος συστήματος ασφαλείας χρησιμοποιεί δύο διαφορετικά κλειδιά για την κρυπτογράφηση των επικοινωνιών μεταξύ δύο μερών.

1. Το ιδιωτικό κλειδί – Αυτό το κλειδί ελέγχεται από τον κάτοχο ενός ιστότοπου και διατηρείται, όπως μπορεί να έχει υποθέσει ο αναγνώστης, ιδιωτικό. Αυτό το κλειδί ζει σε έναν διακομιστή Web και χρησιμοποιείται για την αποκρυπτογράφηση πληροφοριών που κρυπτογραφούνται από το δημόσιο κλειδί.
2. Το δημόσιο κλειδί – Αυτό το κλειδί είναι διαθέσιμο σε όλους όσους θέλουν να αλληλοεπιδράσουν με τον διακομιστή με ασφαλή τρόπο. ΟΙ πληροφορίες που είναι κρυπτογραφημένες από το δημόσιο κλειδί μπορούν να αποκρυπτογραφηθούν μόνο από το ιδιωτικό κλειδί.

2.7.2 TLS

Ο πρωταρχικός στόχος του TLS είναι να παρέχει ένα ασφαλές κανάλι μεταξύ δύο επικοινωνία συνομηλίκων? η μόνη απαίτηση από το υποκείμενο μεταφορά είναι μια αξιόπιστη ροή δεδομένων με τάξη. Συγκεκριμένα, το ασφαλές κανάλι θα πρέπει να παρέχει τις ακόλουθες ιδιότητες:

1. Έλεγχος ταυτότητας: Η πλευρά του διακομιστή του καναλιού είναι πάντα πιστοποιημένος? η πλευρά του πελάτη είναι προαιρετικά έλεγχος ταυτότητας. Ο έλεγχος ταυτότητας μπορεί να πραγματοποιηθεί μέσω ασύμμετρης κρυπτογραφίας (π.χ. RSA , ο αλγόριθμος ψηφιακής υπογραφής ελλειπτικής καμπύλης (ECDSA) ή τον αλγόριθμο ψηφιακής υπογραφής Edwards-Curve (EdDSA) ή ένα συμμετρικό προ-κοινόχρηστο κλειδί (PSK).
2. Εμπιστευτικότητα: Τα δεδομένα που αποστέλλονται μέσω του καναλιού μετά την εγκατάσταση είναι ορατό μόνο στα τελικά σημεία. Το TLS δεν κρύβει το μήκος του τα δεδομένα που μεταδίδει, αν και τα τελικά σημεία μπορούν να συμπληρώσουν το TLS εγγραφές για να αποκρύψουν τα μήκη και να βελτιώσουν την προστασία από τεχνικές ανάλυσης της κυκλοφορίας.
3. Ακεραιότητα: Τα δεδομένα που αποστέλλονται μέσω του καναλιού μετά τη δημιουργία δεν μπορούν να τροποποιηθεί από εισβολείς χωρίς εντοπισμό.

Αυτές οι ιδιότητες θα πρέπει να ισχύουν ακόμη και στο πρόσωπο ενός εισβολέα που έχει τον πλήρη έλεγχο του δικτύου, όπως περιγράφεται στο [RFC3552]. Βλέπω

Το TLS αποτελείται από δύο κύρια στοιχεία:

2. Ένα πρωτόκολλο χειραψίας που πιστοποιεί την ταυτότητα του τα μέρη που επικοινωνούν, διαπραγματεύεται τρόπους κρυπτογράφησης και παραμέτρων και καθιερώνει κοινόχρηστο υλικό πληκτρολόγησης. Το πρωτόκολλο χειραψίας έχει σχεδιαστεί για να αντιστέκεται στην παραβίαση. ενεργός επιτιθέμενος δεν θα πρέπει να είναι σε θέση να αναγκάσει τους ομότιμους να διαπραγματευτούν διαφορετικούς παραμέτρους από ό,τι θα ήταν αν η σύνδεση δεν ήταν κάτω επίθεση.
3. Ένα πρωτόκολλο εγγραφής που χρησιμοποιεί τις καθορισμένες παραμέτρους με το πρωτόκολλο χειραψίας για την προστασία της κυκλοφορίας μεταξύ των συνομηλίκους που επικοινωνούν. Το πρωτόκολλο εγγραφής χωρίζει την

κυκλοφορία σε μια σειρά από αρχεία, καθένα από τα οποία προστατεύεται ανεξάρτητα χρησιμοποιώντας τα κλειδιά κυκλοφορίας.

2.8 Τεχνολογία Blockchain

Το Blockchain είναι μια τεχνολογία ψηφιακού καθολικού που χρησιμοποιεί κρυπτογραφία για την ασφάλεια και την επικύρωση συναλλαγών σε ένα δίκτυο υπολογιστών. Είναι η υποκείμενη τεχνολογία για κρυπτονομίσματα όπως το Bitcoin, αλλά μπορεί επίσης να χρησιμοποιηθεί για μια μεγάλη ποικιλία άλλων εφαρμογών, όπως η διαχείριση της εφοδιαστικής αλυσίδας, τα συστήματα ψηφοφορίας και η ψηφιακή ταυτότητα. Ένα από τα βασικά χαρακτηριστικά του Blockchain είναι ότι είναι αποκεντρωμένο, που σημαίνει ότι δεν ελέγχεται από καμία μεμονωμένη οντότητα και είναι διαφανές, που σημαίνει ότι όλες οι συναλλαγές είναι ορατές σε όλους τους συμμετέχοντες στο δίκτυο. Αυτό το καθιστά εξαιρετικά ανθεκτικό στην παραποίηση και την απάτη και έχει τη δυνατότητα να φέρει επανάσταση σε πολλούς κλάδους.

Το Blockchain λειτουργεί χρησιμοποιώντας ένα δίκτυο υπολογιστών, που ονομάζονται κόμβοι, για την επικύρωση και την καταγραφή συναλλαγών. Όταν πραγματοποιείται μια συναλλαγή, μεταδίδεται σε όλους τους κόμβους του δικτύου και ομαδοποιείται με άλλες συναλλαγές σε ένα μπλοκ. Στη συνέχεια, κάθε μπλοκ συνδέεται κρυπτογραφικά με το προηγούμενο μπλοκ, δημιουργώντας μια αλυσίδα μπλοκ ή μια αλυσίδα μπλοκ. Στη συνέχεια, οι κόμβοι στο δίκτυο συνεργάζονται για να επικυρώσουν τις συναλλαγές στο μπλοκ λύνοντας σύνθετες μαθηματικές εξισώσεις. Μόλις επικυρωθεί ένα μπλοκ, προστίθεται στο blockchain και οι συναλλαγές εντός αυτού γίνονται μόνιμες και αμετάβλητες. Αυτή η διαδικασία είναι γνωστή ως συναίνεση και διασφαλίζει ότι το blockchain είναι ένα ασφαλές και ακριβές αρχείο όλων των συναλλαγών. Εξασφαλίζει επίσης ότι η αποκεντρωμένη φύση του blockchain, καθώς κανένας μεμονωμένος κόμβος ή οντότητα δεν ελέγχει το δίκτυο ή δεν μπορεί να αλλάξει παλαιότερες συναλλαγές.

Υπάρχουν πολλά διαφορετικά δίκτυα blockchain που χρησιμοποιούνται σήμερα, αλλά μερικά από τα πιο γνωστά περιλαμβάνουν το Bitcoin [12], το Ethereum [13]και το Ripple[14].

Το Bitcoin[12], που δημιουργήθηκε το 2009, είναι το πρώτο και πιο ευρέως χρησιμοποιούμενο κρυπτονόμισμα. Χρησιμοποιεί μια αλυσίδα μπλοκ για την καταγραφή και την επικύρωση συναλλαγών που πραγματοποιούνται με Bitcoin. Είναι αποκεντρωμένο, που σημαίνει ότι καμία μεμονωμένη οντότητα δεν το ελέγχει και βασίζεται σε έναν αλγόριθμο συναίνεσης απόδειξης εργασίας.

Το Ethereum[13], που δημιουργήθηκε το 2015, είναι μια πλατφόρμα blockchain που επιτρέπει στους προγραμματιστές να δημιουργούν και να αναπτύσσουν αποκεντρωμένες εφαρμογές, γνωστές ως dapps. Χρησιμοποιεί έναν αλγόριθμο συναίνεσης απόδειξης στοιχήματος και έχει το δικό του κρυπτονόμισμα που ονομάζεται Ether .

Το Ripple[14], που δημιουργήθηκε το 2012, είναι ένα πρωτόκολλο πληρωμών που βασίζεται σε blockchain που επιτρέπει γρήγορες και χαμηλού κόστους διεθνείς μεταφορές χρημάτων. Χρησιμοποιεί έναν μοναδικό αλγόριθμο συναίνεσης που ονομάζεται Αλγόριθμος Συναίνεσης Πρωτοκόλλου Ripple (RPCA) και έχει το δικό του κρυπτονόμισμα που ονομάζεται XRP. Άλλα αξιοσημείωτα blockchain περιλαμβάνουν το Litecoin, το Bitcoin Cash και το EOS, το καθένα έχει τα μοναδικά του χαρακτηριστικά και τον αλγόριθμο συναίνεσης.

2.9 Υπάρχουσα πρόταση

Στο διαδίκτυο υπάρχουν πολλές περιπτώσεις υποκλοπής δεδομένων, ταυτοτήτων και άλλες πολλές επιθέσεις. Έτσι, δημιουργήθηκε πληθώρα πρωτοκόλλων ασφαλής επικοινωνίας όπως είδαμε και πιο πάνω. Στην παρούσα διπλωματική εργασία, αναπτύχθηκε ένα πρωτόκολλο ασφαλής επικοινωνίας το οποίο λαμβάνει υπόψη και

τους πόρους και την ενέργεια που χρειάζεται πράγμα πολύ κρίσιμο διότι λειτουργούμε με μηχανές μικρών πόρων και ελάχιστης κατανάλωσης ενέργειας. Η συγκεκριμένη μεθοδολογία μελετάται αρκετά χρόνια λόγω του ότι θεωρείται κρίσιμη στην αντιμετώπιση κακόβουλων ενεργειών στο δίκτυο.

Συνεπώς, η παρούσα διπλωματική εργασία στηρίχθηκε σε προηγούμενες μελέτες για προσεγγίσεις ασφαλείας αλλά και σε παρόμοιες μελέτες που είδαμε προηγουμένως. Έτσι, μετά από έρευνα καταλήξαμε να δημιουργήσουμε ένα πρωτόκολλο ασφαλείας το οποίο χρησιμοποιεί κλειδιά κρυπτογράφησης και τεχνολογία blockchain έτσι ώστε να διασφαλιστεί η ασφαλής επικοινωνία μεταξύ των μηχανών στο δίκτυο IoT.

Στην διπλωματική αυτή εργασία αναπτύχθηκε ένα ιδιωτικό IoT δίκτυο με 5 μηχανές Raspberry Pi 3 σε τοπολογία Sink-in-the-middle όπου μία από αυτές τις μηχανές είχε το ρόλο του διακομιστή ενώ η άλλες 4 το ρόλο του πελάτη. Χρησιμοποιώντας αλγορίθμους κρυπτογράφησης, κατακερματισμού και χρονικών σφραγίδων που θα δούμε στη συνέχεια επικοινωνούν μεταξύ τους.

Σημαντικό είναι ότι στην παρούσα διπλωματική εργασία σημαντικό ρόλο εκτός από την ασφαλή επικοινωνία μεταξύ των διαφόρων μηχανών έχει και απόδοση της επικοινωνίας αυτής, καθώς λήφθηκε υπόψη και ο χρόνος που χρειάζεται το πρωτόκολλο που υλοποιήθηκε σε σχέση με τα υφιστάμενα πρωτοκολλά. Παράλληλα, εκτελέστηκαν και πειράματα όπου προσπαθήσαμε να ελέγξουμε διάφορα είδη επιθέσεων και να καταγράψουμε πως το πρωτόκολλο μας αντεπεξέρχεται στις επιθέσεις αυτές.

Κεφάλαιο 3
Επιθέσεις

Αυτή η ενότητα δείχνει τις επιθέσεις στο IoT από τις πιο επικίνδυνες έως τις λιγότερο επικίνδυνες.

3.1 Επισκόπηση

Οι επιθέσεις που εμφανίζονται στο Διαδίκτυο των Πραγμάτων, μπορεί να προέρχονται από τα κανάλια που συνδέουν τα στοιχεία μεταξύ τους. Αρκετά πρωτόκολλα που χρησιμοποιούνται σε συσκευές του Διαδικτύου των Πραγμάτων τυγχάνει να έχουν ζητήματα ασφαλείας που μπορεί να επηρεάσουν ολόκληρο το δίκτυο. Επίσης, οι συσκευές αυτές είναι ευαίσθητες σε επιθέσεις δικτύου που επηρεάζουν την εμπιστευτικότητα και την ακεραιότητα των πληροφοριών που μεταδίδονται από τη μία προς την άλλη. Σε αυτό το κεφάλαιο, παρουσιάζεται μια γκάμα από επιθέσεις που χρησιμοποιήθηκαν για την αξιολόγηση του συγκεκριμένου πρωτοκόλλου ασφαλείς επικοινωνίας και συχνά εμφανίζονται στο Διαδίκτυο των Πραγμάτων.

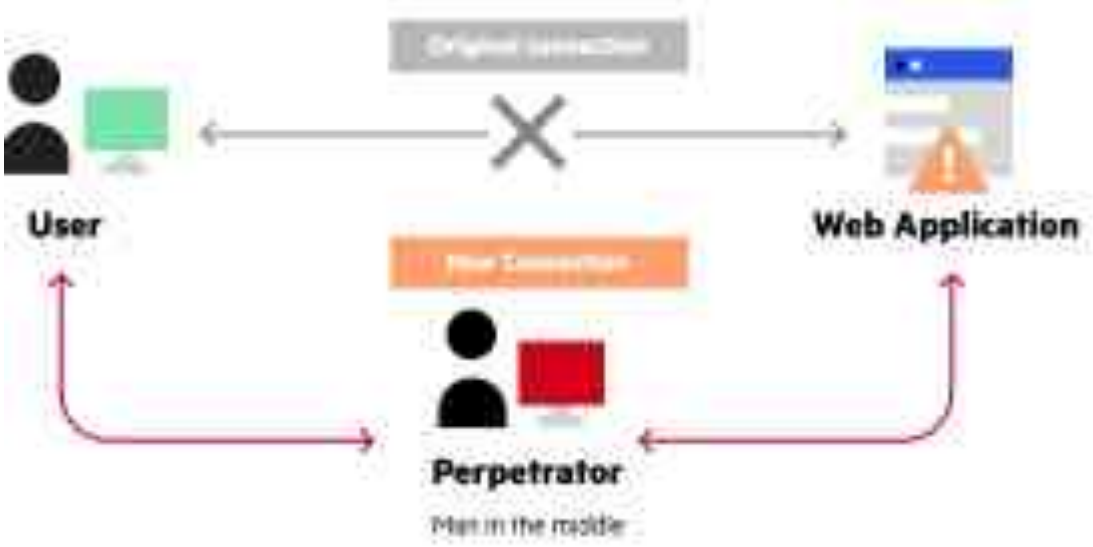
3.2 Man-in-the-Middle

Η επίθεση ¨Man in the Middle¨ είναι ένας γενικός όρος για όταν ένας δράστης τοποθετείται σε μια συνομιλία μεταξύ ενός χρήστη και μιας εφαρμογής, είτε για να κρυφακούει είτε για να υποδυθεί ένα από τα μέρη, κάνοντας το να φαίνεται σαν μια κανονική ανταλλαγή πληροφοριών.

Ο στόχος μιας επίθεσης είναι να κλέψει προσωπικές πληροφορίες, όπως διαπιστευτήρια σύνδεσης, στοιχεία λογαριασμού και αριθμούς πιστωτικών καρτών. Οι στόχοι είναι συνήθως οι χρήστες οικονομικών εφαρμογών, επιχειρήσεις SaaS, ισότοποι ηλεκτρονικού εμπορίου και άλλοι ιστότοποι όπου απαιτείται σύνδεση.

Οι πληροφορίες που λαμβάνονται κατά τη διάρκεια μιας επίθεσης θα μπορούσαν να χρησιμοποιηθούν για πολλούς σκοπούς, συμπεριλαμβανομένης κλοπής ταυτότητας, των μη εγκεκριμένων μεταφορών χρημάτων ή μιας παράνομης αλλαγής κωδικού πρόσβασης.

Επιπλέον, μπορεί να χρησιμοποιηθεί για να αποκτήσει πρόσβαση μέσα σε μια ασφαλή περίμετρο κατά τη διάρκεια του σταδίου διείσδυσης μια επίθεσης προηγμένης απειλής (APT). Σε γενικές γραμμές μια επίθεση ¨Man in the Middle¨ (Σχήμα 7), ισοδυναμεί με έναν ταχυδρόμο που ανοίγει την κίνηση του τραπεζικού λογαριασμού σας, σημειώνει τα στοιχεία του λογαριασμού σας και στη συνέχεια σφραγίζει ξανά τον φάκελο και τον παραδίδει στην πόρτα σας.



Σχήμα 7

3.3 Replay Attack

Μια επίθεση επανάληψης είναι μια επίθεση κατά την οποία ο επιτιθέμενος καταγράφει την συνεδρία επικοινωνίας και επαναλαμβάνει ολόκληρη την συνεδρία, ή κάποιο μέρος της περιόδου λειτουργίας, σε μεταγενέστερο χρονικό σημείο. Τα επαναλαμβανόμενα μηνύματα μπορούν να σταλούν στον ίδιο επαληθευτή με αυτόν που συμμετείχε στην αρχική περίοδο λειτουργίας ή σε διαφορετικό επαληθευτή.

Ο στόχος της επίθεσης επανάληψης μπορεί να είναι η πλαστοπροσωπία ή μπορεί να είναι κάποια άλλη εξαπάτηση (π.χ μια επιτυχημένη ανταλλαγή πρωτοκόλλου για μεταφορά χρημάτων από τον λογαριασμό του Α στον λογαριασμό του Β, μπορεί να επαναληφθεί από τον Β σε μια προσπάθεια να μεταφέρει περισσότερα χρήματα από ότι είχε σκοπό ο Α) [7]

3.4.1 DDoS Attacks

Όταν οι χάκερ συνενώνονται για να καταργήσουν ένα ιστότοπο, συχνά το κάνουν χρησιμοποιώντας μια επίθεση κατανεμημένης άρνησης υπηρεσίας (DDoS).

Εκατομμύρια ιστότοποι έχουν περιορισμένους πόρους διακομιστή ιστού και δικτύου για τη διαχείριση της κυκλοφορίας. Για παράδειγμα, ένα μικρό τοπικό αρτοποιείο θα είχε έναν ιστότοπο που χρειάζεται μόνο να εξυπηρετεί μερικές εκατοντάδες επισκέπτες το μήνα. Και με περιορισμένα κεφάλαια διαθέσιμα για ξόδεμα στη φιλοξενία Ιστού, ο ιδιοκτήτης μπορεί να δημιουργήσει έναν ιστότοπο χρησιμοποιώντας κοινόχρηστη φιλοξενία που δεν μπορεί να χειριστεί πολύ περισσότερους από αυτές τις λίγες εκατοντάδες επισκέπτες.

Εάν υπήρχε ξαφνική αύξηση της επισκεψιμότητας, ο ιστότοπος θα υπερφορτωθεί και θα κατέρρεε - καθιστώντας τον άχρηστο για όλους. Πρόκειται ουσιαστικά για επίθεση DDoS.

Μπορείτε να δείτε γιατί ένας ιστότοπος μικρών επιχειρήσεων θα ήταν πιο ευάλωτος σε αυτού του είδους τις επιθέσεις. Θα χρειαζόταν πολλούς πόρους για να κατακλύσει το σύστημα, για παράδειγμα, στο Facebook. Αλλά αυτό το αρτοποιείο θα ήταν εύκολο να διαλέξετε.

3.4.2 Goldeneye Attack

Το Goldeneye είναι ένα δωρεάν εργαλείο ανοιχτού κώδικα διαθέσιμο στο GitHub. Μπορούμε να εκτελέσουμε μια επίθεση άρνησης υπηρεσίας χρησιμοποιώντας αυτό το εργαλείο. Είναι ένα πλαίσιο γραμμένο σε .NET Core. Αυτό το εργαλείο παρέχει πολλές βασικές κλάσεις και επεκτάσεις για χρήση με την καθημερινή σας εργασία. Αυτό το

εργαλείο επιτρέπει σε ένα μόνο μηχάνημα να καταργήσει τον διακομιστή ιστού άλλου μηχανήματος και χρησιμοποιεί απόλυτα νόμιμη κυκλοφορία HTTP. Κάνει μια πλήρη σύνδεση TCP και στη συνέχεια απαιτεί μόνο μερικές εκατοντάδες αιτήματα σε μακροπρόθεσμα και τακτικά διαστήματα. Ως αποτέλεσμα, το εργαλείο δεν χρειάζεται να χρησιμοποιεί πολλή κίνηση για να εξαντλήσει τις διαθέσιμες συνδέσεις σε έναν διακομιστή.

Χρήσεις του Goldeneye:

Το Goldeneye χρησιμοποιεί απόλυτα νόμιμη κίνηση HTTP.

Η επίθεση άρνησης υπηρεσίας μπορεί να εκτελεστεί με τη βοήθεια του Goldeneye δημιουργώντας μεγάλη κίνηση από botnet.

Το Goldeneye στέλνει πολλαπλά αιτήματα στον στόχο με αποτέλεσμα να δημιουργεί botnet βαριάς κυκλοφορίας.

Το Goldeneye είναι ένα εργαλείο ανοιχτού κώδικα, ώστε να μπορείτε να το κατεβάσετε από το GitHub δωρεάν.

Το Goldeneye μπορεί να χρησιμοποιηθεί για την εκτέλεση επιθέσεων ddos σε οποιονδήποτε διακομιστή ιστού.

3.4.3 HTTP Attack

Μια επίθεση DDoS πλημμύρας HTTP χρησιμοποιεί ό,τι φαίνεται να είναι νόμιμα αιτήματα HTTP GET ή POST για να επιτεθεί σε έναν διακομιστή ιστού ή μια εφαρμογή. Αυτές οι πλημμυρικές επιθέσεις DDoS βασίζονται συχνά σε ένα botnet, το οποίο είναι μια ομάδα υπολογιστών συνδεδεμένων στο Διαδίκτυο που έχουν οικειοποιηθεί κακόβουλα μέσω της χρήσης κακόβουλου λογισμικού όπως ο Δούρειος Ίππος.

3.5.1 Brute Force

Το Brute Force Attack είναι η πιο συμβατική επίθεση που δρα ενάντια σε εφαρμογές ιστού. Αποκτάς πρόσβαση στους λογαριασμούς χρηστών προσπαθώντας να μαντέψεις το κωδικό πρόσβασης του μεμονωμένου χρήστη ή ομάδας χρηστών είναι ο βασικός στόχος του Brute Force Attack. Η εφαρμογή Ιστού θα πρέπει να είναι αρκετά στιβαρή

ώστε να μην λειτουργεί αυτή η επίθεση, εκτός εάν ο εισβολέας αποκτήσει το προνόμιο του συστήματος.

Οι επιθέσεις Brute Force μπορούν να εφαρμοστούν με πολλούς τρόπους. Αν το μήκος του κωδικού πρόσβασης είναι γνωστό από τον εισβολέα μπορεί να προκαλέσει την Brute Force Attack, με συνδυασμό αριθμών, γραμμάτων και συμβόλων μπορούν να εφαρμοστούν εκτός εάν υπάρχει κατάλληλη αντιστοίχιση βρέθηκαν. Ωστόσο, αυτή είναι μια αργή διαδικασία, ειδικά καθώς το μήκος του κωδικού πρόσβασης αυξάνεται. Ένας από τους τρόπους του Brute Force Attack είναι εάν ο επιτιθέμενος γνωρίζει το όνομα χρήστη που είναι γενικά root για μια εφαρμογή ιστού. Περαιτέρω μπορεί να βασίζεται στην πολυπλοκότητα του κωδικού πρόσβασης, για παράδειγμα αν χρησιμοποιείται ένα αδύναμο κωδικό πρόσβασης, τότε γίνεται θύμα επίθεσης.

3.5.2 Dictionary Attack

Η επίθεση λεξικού είναι μια μέθοδος διάρρηξης ενός υπολογιστή, δικτύου ή άλλου πόρου πληροφορικής που προστατεύεται με κωδικό πρόσβασης, εισάγοντας συστηματικά κάθε λέξη ενός λεξικού ως κωδικό πρόσβασης. Μια επίθεση λεξικού μπορεί επίσης να χρησιμοποιηθεί σε μια προσπάθεια να βρεθεί το κλειδί που είναι απαραίτητο για την αποκρυπτογράφηση ενός κρυπτογραφημένου μηνύματος ή εγγράφου.

Οι επιθέσεις λεξικών λειτουργούν επειδή πολλοί χρήστες υπολογιστών και επιχειρήσεις επιμένουν να χρησιμοποιούν συνηθισμένες λέξεις ως κωδικούς πρόσβασης. Αυτές οι επιθέσεις είναι συνήθως ανεπιτυχείς σε συστήματα που χρησιμοποιούν κωδικούς πρόσβασης πολλαπλών λέξεων και είναι επίσης συχνά ανεπιτυχείς έναντι κωδικών πρόσβασης που αποτελούνται από κεφαλαία και πεζά γράμματα και αριθμούς σε τυχαίους συνδυασμούς.

Σε συστήματα με ισχυρές απαιτήσεις κωδικού πρόσβασης, η μέθοδος επίθεσης brute-force, στην οποία κάθε πιθανός συνδυασμός χαρακτήρων και διαστημάτων δοκιμάζεται μέχρι ένα ορισμένο μέγιστο μήκος, μπορεί μερικές φορές να είναι αποτελεσματική. Ωστόσο, μια Brute Force Attack μπορεί να πάρει πολύ χρόνο για να δώσει αποτελέσματα.

Οι ισχυροί, τυχαιοποιημένοι κωδικοί πρόσβασης δεν μπορούν να προβλεφθούν εύκολα και είναι πολύ απίθανο να συμπεριληφθούν στην προκαθορισμένη βιβλιοθήκη κωδικών

πρόσβασης. Επειδή οι προσπάθειες εικασίας μιας επίθεσης λεξικού περιορίζονται σε μια προεπιλεγμένη λίστα, είναι ουσιαστικά αδύνατο να σπάσετε μη προβλέψιμους κωδικούς πρόσβασης.

3.6 SQL Injection

Μια επίθεση SQL injection αποτελείται από την εισαγωγή ή την "ένεση" ενός ερωτήματος SQL μέσω των δεδομένων εισόδου από τον πελάτη στην εφαρμογή. Μια επιτυχημένη εκμετάλλευση έγχυσης SQL μπορεί να διαβάσει ευαίσθητα δεδομένα από τη βάση δεδομένων, να τροποποιήσει δεδομένα βάσης δεδομένων (Εισαγωγή/Ενημέρωση/Διαγραφή), να εκτελέσει λειτουργίες διαχείρισης στη βάση δεδομένων (όπως ο τερματισμός λειτουργίας του DBMS), να ανακτήσει το περιεχόμενο ενός δεδομένου αρχείου που υπάρχει στο αρχείο DBMS σύστημα και σε ορισμένες περιπτώσεις εκδίδει εντολές στο λειτουργικό σύστημα. Οι επιθέσεις SQL injection είναι ένας τύπος επίθεσης injection, στην οποία οι εντολές SQL εισάγονται στην είσοδο του επιπέδου δεδομένων προκειμένου να επηρεάσουν την εκτέλεση προκαθορισμένων εντολών SQL.

Οι επιθέσεις SQL Injection επιτρέπουν στους εισβολείς να πλαστογραφήσουν την ταυτότητα, να παραποιήσουν υπάρχοντα δεδομένα, να προκαλέσουν ζητήματα απόρριψης όπως ακύρωση συναλλαγών ή αλλαγή υπολοίπων, να επιτρέπουν την πλήρη αποκάλυψη όλων των δεδομένων στο σύστημα, να καταστρέψουν τα δεδομένα ή να τα καταστήσουν διαφορετικά διαθέσιμα και να γίνουν διαχειριστές του διακομιστή βάσης δεδομένων.

Το SQL Injection είναι πολύ κοινό με τις εφαρμογές PHP και ASP λόγω της επικράτησης παλαιότερων λειτουργικών διεπαφών. Λόγω της φύσης των διαθέσιμων διεπαφών μέσω προγραμματισμού, οι εφαρμογές J2EE και ASP.NET είναι λιγότερο πιθανό να έχουν εύκολα εκμεταλλευόμενες ενέσεις SQL.

Η σοβαρότητα των επιθέσεων SQL Injection περιορίζεται από την ικανότητα και τη φαντασία του εισβολέα και, σε μικρότερο βαθμό, από τα αντίμετρα άμυνας σε βάθος, όπως συνδέσεις χαμηλών προνομίων με τον διακομιστή της βάσης δεδομένων και ούτω καθεξής. Σε γενικές γραμμές, θεωρήστε το SQL Injection μια υψηλή σοβαρότητα επιπτώσεων.

Κεφάλαιο 4
Μεθοδολογία

4.1 Επισκόπηση

Στην παρούσα διπλωματική εργασία υλοποιείται ένα πρωτόκολλο επικοινωνίας για μηχανές στο Διαδίκτυο των Πραγμάτων. Στο πρωτόκολλο αυτό χρησιμοποιήθηκαν διάφορες τεχνικές κρυπτογράφησης όπως AES και RSA σε συνδυασμό με την τεχνολογία blockchain καθώς και άλλες τεχνικές όπως η χρονομέτρηση των πακέτων έτσι ώστε να επιτευχθεί ασφαλείς επικοινωνία μεταξύ των μηχανών. Σε αυτή την έρευνα, όπως ανάφερα και προηγουμένως αναπτύχθηκε ένα πρωτόκολλο ασφαλής επικοινωνίας συσκευών στο Διαδίκτυο των Πραγμάτων χρησιμοποιώντας την γλώσσα προγραμματισμού java (μπορείτε να βρείτε τον κώδικα εδώ [15]) το οποίο εφαρμόστηκε σε 5 Raspberry Pie3 μηχανές χρησιμοποιώντας Sink-in-the-Middle τοπολογία παρόλο που δουλεύει σε όλες τις τοπολογίες οι οποίες χρησιμοποιούν Sink node.

4.2 Ανάλυση ασφάλειας του πρωτοκόλλου μας και επίσημη απόδειξη ασφαλείας

Σε αυτήν την ενότητα, συζητάμε ζητήματα ασφάλειας που μπορεί να προκύψουν με τη χρήση του προτεινόμενου πρωτοκόλλου σύμφωνα με τους στόχους ασφαλείας μας και να εξετάσουμε εάν και πώς η προσέγγισή μας ικανοποιεί τον καθένα από αυτούς τους στόχους.

Blockchain:

Ένα blockchain είναι η ραχοκοκαλιά της Τεχνολογίας Ψηφιακών Κρυπτονομισμάτων (για παράδειγμα Bitcoin)

• Μια αλυσίδα μπλοκ είναι μια λίστα εγγραφών που ονομάζονται μπλοκ που συνδέονται μεταξύ τους χρησιμοποιώντας συνδεδεμένες λίστες και μερικών αλγόριθμων κρυπτογράφησης .

• Κάθε μπλοκ περιέχει το δικό του ψηφιακό αποτύπωμα που ονομάζεται Hash, το κατακερματισμό του προηγούμενου μπλοκ, τη χρονική σήμανση και τα δεδομένα της συναλλαγής που έγινε, την καθιστούν πιο ασφαλή έναντι κάθε είδους παραβίαση δεδομένων.

• Επομένως, εάν αλλάξουν τα δεδομένα ενός μπλοκ, τότε θα αλλάξει και ο κατακερματισμός του. Αν ο κατακερματισμός αλλάξει, τότε ο κατακερματισμός του θα είναι διαφορετικός από το επόμενο μπλοκ που περιέχει τον κατακερματισμό του προηγούμενο μπλοκ που επηρεάζει όλους τους κατακερματισμούς των μπλοκ μετά από αυτό. Αλλαγή των κατακερματισμών και μετά συγκρίνοντάς το με άλλα μπλοκ μας επιτρέπει να ελέγξουμε το blockchain.

• Δημιουργία κατακερματισμού: Για τη δημιουργία κατακερματισμού, χρησιμοποιείται ο αλγόριθμος SHA256.

• Εγκυρότητα Blockchain: Τέλος, πρέπει να ελέγξουμε την εγκυρότητα του Blockchain, δημιουργώντας μια Boolean μέθοδο για τον έλεγχο της εγκυρότητας. Αυτή η μέθοδος θα εφαρμοστεί στον διακομιστή και ελέγχει εάν ο κατακερματισμός είναι ίσος με τον υπολογισμένο κατακερματισμό ή όχι. Αν όλοι οι κατακερματισμοί είναι ίσοι με τους υπολογισμένους κατακερματισμούς, τότε το μπλοκ είναι έγκυρο.

4.3.1 Διαδικασία Εγγραφής Συσκευών

Αρχικά, ο πελάτης δημιουργεί ένα συμμετρικό κλειδί AES. Έπειτα, ζητά από τον διακομιστή να του δώσει το ασύμμετρο δημόσιο του κλειδί RSA. Στη συνέχεια, ο πελάτης κρυπτογραφεί τα δεδομένα του χρησιμοποιώντας το συμμετρικό του κλειδί AES, το οποίο κλειδί στη συνέχεια το κρυπτογραφεί χρησιμοποιώντας το ασύμμετρο δημόσιο κλειδί RSA του διακομιστή. Μετέπειτα στέλνει το κρυπτογραφημένο συμμετρικό κλειδί του και τα κρυπτογραφημένα του δεδομένα εγγραφής στον διακομιστή.

Στη συνέχεια, ο διακομιστής αποκρυπτογραφεί το συμμετρικό κλειδί AES του πελάτη χρησιμοποιώντας το ασύμμετρο ιδιωτικό του κλειδί RSA. Έπειτα, με το αποκρυπτογραφημένο πλέον ασύμμετρο κλειδί AES του πελάτη, αποκρυπτογραφεί τα δεδομένα που του έστειλε ο πελάτης. Αργότερα, ελέγχει την χρονική σήμανση (timestamp) του πακέτου για αποφυγή των επιθέσεων επανάληψης. Από τα δεδομένα που παίρνει το hash και ελέγχει αν υπάρχει μέσα στο blockchain.

Αν δεν υπάρχει ήδη μέσα στο blockchain, δημιουργεί ένα νέο block μέσα στο blockchain και συσχετίζει το id του block με το hash που πήρε από το πελάτη και έτσι ο πελάτης εγγράφηκε με επιτυχία.

Από την άλλη όμως, αν υπάρχει ήδη το hash μέσα στο blockchain, τότε δεν είναι δυνατή η εγγραφή. Έτσι στέλνεται ένα μήνυμα στον πελάτη ότι η εγγραφή δεν είναι δυνατή διότι το hash υπάρχει ήδη μέσα στο blockchain και η σύνδεση διακόπτεται.

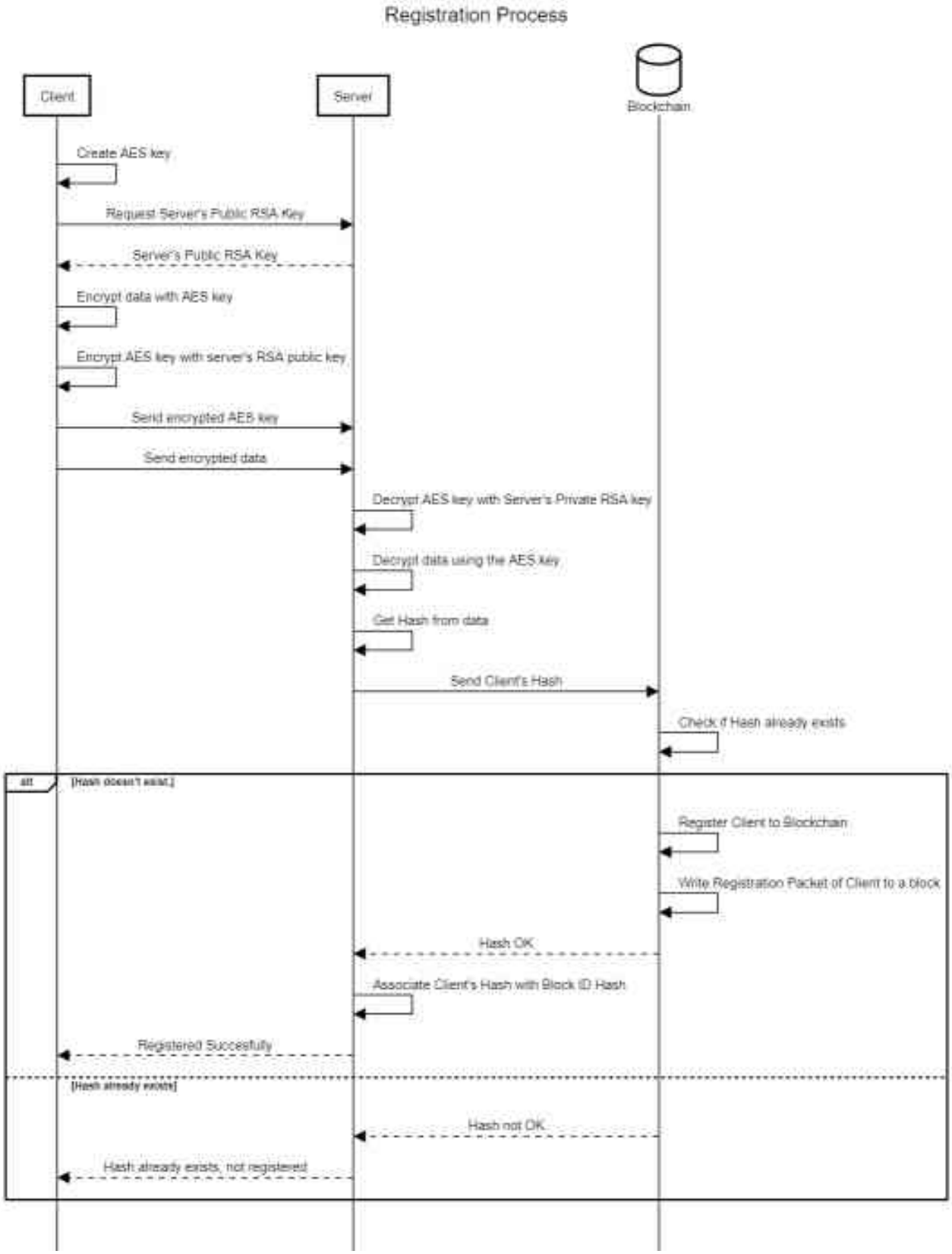
Αξιοσημείωτο το γεγονός πως κάθε 30 μέρες όλα τα κλειδιά συμμετρικά ή ασύμμετρα αλλάζουν για επιπλέον ασφάλεια.

4.3.2 Πακέτο εγγραφής

Το πακέτο εγγραφής περιέχει κάποια στοιχεία για το χρήστη και τη μηχανή, όπως το MAC Address της μηχανής, την IP διεύθυνση της μηχανής, όνομα μηχανής, όνομα χρήστη και κωδικό πρόσβασης. Στη συνέχεια το πακέτο υπογράφετε με το ασύμμετρο ιδιωτικό RSA κλειδί του χρήστη και υπολογίζεται η τιμή κατακερματισμού με τη χρήση SHA-256. Έπειτα, δημιουργείται ένα αντίγραφο του πακέτου αυτού το οποίο περιέχει και την τιμή κατακερματισμού και αυτό το πακέτο είναι το πακέτο που στέλνετε στον διακομιστή στην διαδικασία εγγραφής. Κάθε πακέτο εγγραφής ανανεώνεται κάθε 30 μέρες, όπου επιλέγεται ένα νέο τυχαίο όνομα χρήστη και ένα νέος τυχαίος κωδικός. Τα στοιχεία του πακέτου εγγραφής αναπαρίστανται στα πιο κάτω Σχήματα 8 1 και 8 2, ενώ η διαδικασία εγγραφής στο Σχήμα 9.

Σχήμα 8 1

Σχήμα 8 2


Registration Process
Client
Server
Blockchain
Create AES key
Request Server's Public RSA Key
Server's Public RSA Key
Encrypt data with AES key
Encrypt AES key with server's RSA public key
Send encrypted AES key
Send encrypted data
Decrypt AES key with Server's Private RSA key
Decrypt data using the AES key
Get Hash from data
Send Client's Hash
Check if Hash already exists
alt
[Hash doesn't exist]
Register Client to Blockchain
Write Registration Packet of Client to a block
Hash OK
Associate Client's Hash with Block ID Hash
Registered Succesfully
[Hash already exists]
Hash not OK
Hash already exists, not registered

Σχήμα 9

4.4.1 Διαδικασία Ταυτοποίησης Συσκευών

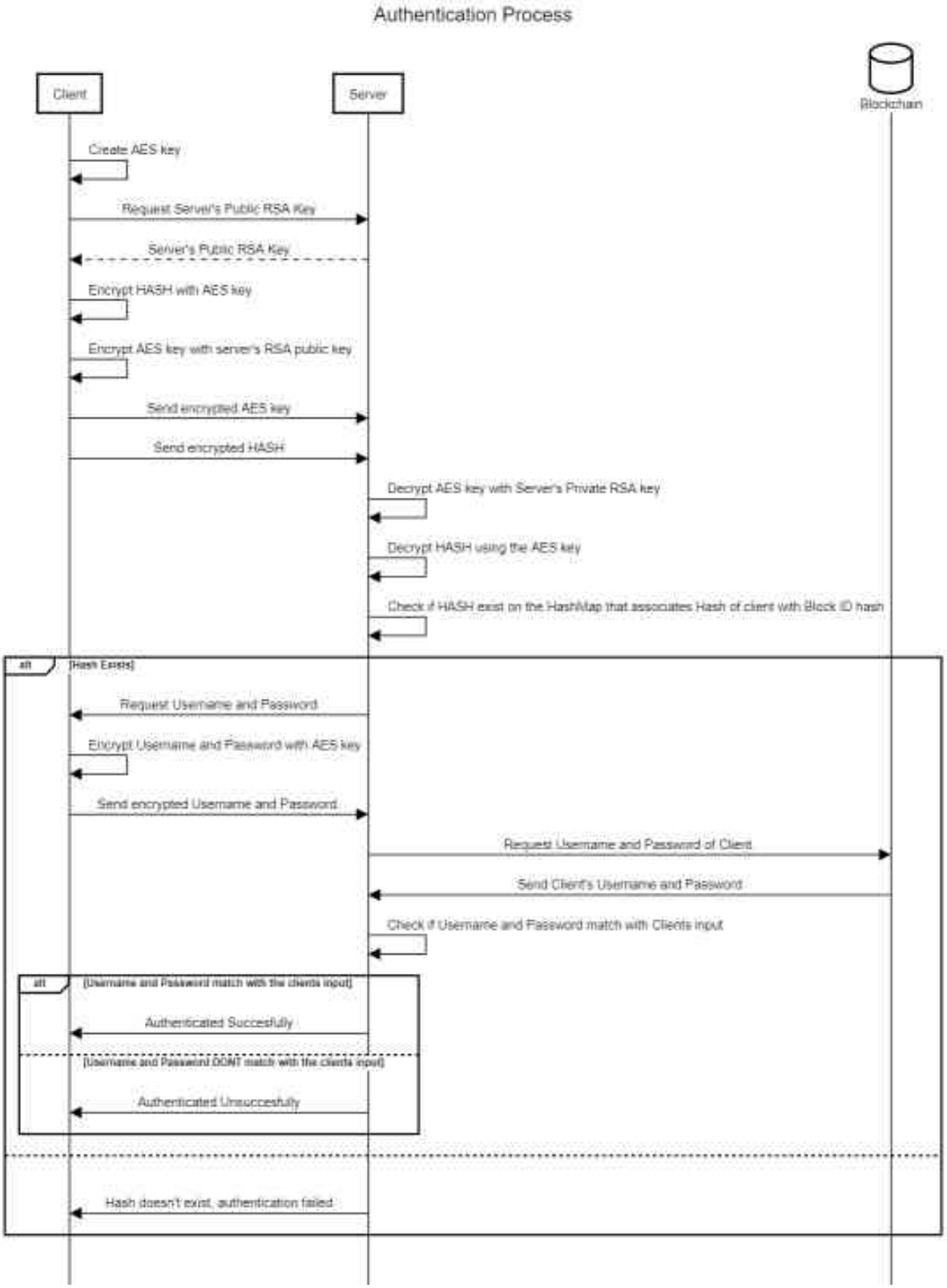
Αρχικά, ο πελάτης δημιουργεί ένα συμμετρικό κλειδί AES. Έπειτα, ζητά από τον διακομιστή να του δώσει το ασύμμετρο δημόσιο του κλειδί RSA. Στη συνέχεια, ο πελάτης κρυπτογραφεί τα δεδομένα του χρησιμοποιώντας το συμμετρικό του κλειδί AES, το οποίο κλειδί στη συνέχεια το κρυπτογραφεί χρησιμοποιώντας το ασύμμετρο δημόσιο κλειδί RSA του διακομιστή. Μετέπειτα στέλνει το κρυπτογραφημένο συμμετρικό κλειδί του και τα κρυπτογραφημένα του δεδομένα ταυτοποίησης στον διακομιστή.

Στη συνέχεια, ο διακομιστής αποκρυπτογραφεί το συμμετρικό κλειδί AES του πελάτη χρησιμοποιώντας το ασύμμετρο ιδιωτικό του κλειδί RSA. Έπειτα, με το αποκρυπτογραφημένο πλέον ασύμμετρο κλειδί AES του πελάτη, αποκρυπτογραφεί τα δεδομένα που του έστειλε ο πελάτης. Αργότερα, ελέγχει την χρονική σήμανση (timestamp) του πακέτου για αποφυγή των επιθέσεων επανάληψης. Από τα δεδομένα που παίρνει το hash και ελέγχει αν υπάρχει μέσα στο blockchain.

Αν υπάρχει στο blockchain, τότε ο διακομιστής ζητά από τον πελάτη όνομα χρήστη και κωδικό. Στη συνέχεια, ο πελάτης αποκρυπτογραφεί το όνομα χρήστη και το κωδικό του και τα στέλνει στο διακομιστή. Τέλος, ελέγχονται τα στοιχεία που μόλις έστειλε ο πελάτης (όνομα χρήστη και κωδικό) με τα στοιχεία που έχει το μπλοκ στο blockchain και αν είναι τα ίδια τότε η ταυτοποίηση είναι επιτυχής αλλιώς αποτυγχάνει και η σύνδεση τερματίζεται.

Αν δεν υπάρχει στο blockchain, τότε η ταυτοποίηση δεν είναι επιτυχής και η σύνδεση τερματίζεται.

Αξιοσημείωτο το γεγονός πως κάθε 30 μέρες όλα τα κλειδιά συμμετρικά ή ασύμμετρα αλλάζουν για επιπλέον ασφάλεια. Η διαδικασία ταυτοποίησης αναπαρίσταται στο πιο κάτω Σχήμα 10.

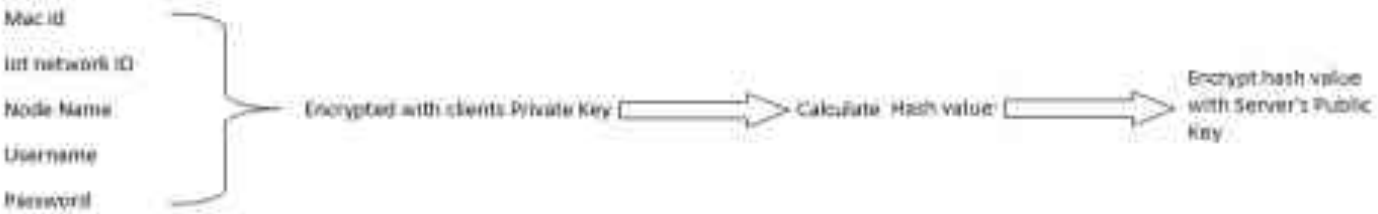

Authentication Process
Client
Server
Blockchain
Create AES key
Request Server's Public RSA Key
Server's Public RSA Key
Encrypt HASH with AES key
Encrypt AES key with server's RSA public key
Send encrypted AES key
Send encrypted HASH
Decrypt AES key with Server's Private RSA key
Decrypt HASH using the AES key
Check if HASH exist on the HashMap that associates Hash of client with Block ID hash
alt
[Hash Exists]
Request Username and Password
Encrypt Username and Password with AES key
Send encrypted Username and Password
Request Username and Password of Client
Send Client's Username and Password
Check if Username and Password match with Clients input
alt
[Username and Password match with the clients input]
Authenticated Succesfully
[Username and Password DONT match with the clients input]
Authenticated Unsuccesfully
Hash doesn't exist, authentication failed

Σχήμα 10

4.4.2 Πακέτο Ταυτοποίησης

Το πακέτο ταυτοποίησης περιέχει κάποια στοιχεία για το χρήστη και τη μηχανή, όπως το MAC Address της μηχανής, την IP διεύθυνση της μηχανής, όνομα μηχανής, όνομα χρήστη και κωδικό πρόσβασης. Στη συνέχεια το πακέτο υπογράφετε με το ασύμμετρο ιδιωτικό RSA κλειδί του χρήστη και υπολογίζεται η τιμή κατακερματισμού με τη χρήση SHA-256. Έπειτα, στέλνεται η κρυπτογραφημένη τιμή κατακερματισμού στον διακομιστή, ο οποίος μόλις συσχετίσει την τιμή κατακερματισμού με ένα μπλοκ στο blockchain παίρνει τα στοιχεία του μπλοκ που ουσιαστικά είναι το πακέτο εγγραφής κάθε πελάτη και ζητάει από τον πελάτη το όνομα χρήστη και κωδικό πρόσβασης για διπλή ταυτοποίηση. Κάθε πακέτο ταυτοποίησης ανανεώνεται κάθε 30 μέρες, όπου επιλέγεται ένα νέο τυχαίο όνομα χρήστη και ένα νέος τυχαίος κωδικός. Τα στοιχεία του πακέτου ταυτοποίησης αναπαρίστανται στα πιο κάτω Σχήματα 11 1 και 11 2.

Σχήμα 11 2



Σχήμα 11 1

4.5 Blockchain

Στη παρούσα διπλωματική εργασία αναπτύχθηκε ένα σχετικά απλό αλλά παράλληλα πολύ ισχυρό blockchain. Υλοποιήθηκε με βάση τη λογική του blockchain δηλαδή ότι κάθε μπλοκ είναι συνδεδεμένο με το προηγούμενο καθώς και το επόμενο του μπλοκ στην αλυσίδα και πως με οποιαδήποτε αλλαγή σε οποιοδήποτε μπλοκ τότε η αλυσίδα σπάζει και toblockchain δεν μπορεί να θεωρηθεί πλέον αξιόπιστο, ένα μπλοκ όπως φαίνεται και στο πιο κάτω σχεδιάγραμμα αποτελείται από τα δεδομένα που έχει μέσα την τιμή κατακερματισμού του προηγούμενου μπλοκ, την χρονική στιγμή που δημιουργήθηκε και την τιμή κατακερματισμού του. η τιμή κατακερματισμού κάθε μπλοκ υπολογίζεται βάσει την τιμή κατακερματισμού του προηγούμενου μπλοκ σε συνδυασμό με την χρονική στιγμή που δημιουργήθηκε το μπλοκ και τα δεδομένα του μπλοκ. Έτσι, ο οποιοσδήποτε προσπαθήσει να ¨πειράξει¨, μεταβάλλει τα δεδομένα που υπάρχουν στο blockchain, αυτόματα θα αλλάξει και τις τιμές κατακερματισμού των μπλοκ, πράγμα που κάνει την αλυσίδα να σπάζει αφού βασίζεται στις τιμές αυτές για να μείνει ενωμένη.

Αξιοσημείωτο το γεγονός ότι πριν και μετά από κάθε νέα σύνδεση ελέγχεται η αλυσίδα. Μία αναπαράσταση της Κλάσης Block που χρησιμοποιείται για τις διάφορες λειτουργίες της αλυσίδας φαίνεται στο πιο Κάτω Σχήμα 12.

Σχήμα 12

4.6 Διαδικασία δημιουργίας νέας αλυσίδας

Το blockchain δεν μπλοκάρει ούτε σπάζει ποτέ, είναι εγγυημένο ότι είναι στο διαδίκτυο και αμετάβλητο, είναι η πιο ασφαλή δομή δεδομένων στην πληροφορική μέχρι τώρα, αναπαράγεται και έχει περιπτώσεις master και slave. Λόγω του ότι δεν χρησιμοποιήσαμε etherrum ή άλλο ευρέως γνωστή βιβλιοθήκη υπάρχει περίπτωση να σπάσει το μπλοκ στην αλυσίδα λόγω αδυναμίας της βιβλιοθήκης να αντιμετωπίσει αυτό το θέμα. Σε περίπτωση που η αλυσίδα "σπάσει", διακόπτονται όλες οι συνδέσεις όλα τα δεδομένα διαγράφονται και όλα τα κλειδιά ανανεώνονται σε πρώτο στάδιο. Πρέπει να το μετρήσω, χρόνος να σβηστούν και να δημιουργηθούν.

4.6.1 Αποθήκευση δεδομένων
Blockchain

Για την αποθήκευση της αλυσίδας χρησιμοποιήθηκε η δομή δεδομένων "array list" η οποία περιέχει το αντικείμενο μπλοκ συνδέοντας το κάθε στοιχείο της λίστας όπως συνδέονται μεταξύ τους τα μπλοκ, όπως παρουσιάστηκε πιο πάνω. Σε περίπτωση τερματισμού του διακομιστή η αλυσίδα φυλάσσεται σε ένα αρχείο ονόματι "blockchain.dat" με την χρήση της κλάσης "serialazable".

Κλειδιά

Όλα τα συμμετρικά και ασύμμετρα κλειδιά δημιουργούνται και φυλάσσονται με την χρήση της βιβλιοθήκης «crypto» της java. Παρέχει τις κλάσεις και τις διεπαφές για κρυπτογραφικές λειτουργίες. Οι κρυπτογραφικές λειτουργίες που ορίζονται σε αυτό το πακέτο περιλαμβάνουν κρυπτογράφηση, δημιουργία κλειδιού και συμφωνία κλειδιού και δημιουργία κωδικού ελέγχου ταυτότητας μηνυμάτων (MAC).
Η υποστήριξη για κρυπτογράφηση περιλαμβάνει συμμετρικούς, ασύμμετρους, μπλοκ και κρυπτογράφηση ροής. Αυτό το πακέτο υποστηρίζει επίσης ασφαλείς ροές και σφραγισμένα αντικείμενα.

Πολλές από τις κατηγορίες που παρέχονται σε αυτό το πακέτο βασίζονται σε πάροχους. Η ίδια η κλάση ορίζει μια διεπαφή προγραμματισμού στην οποία μπορούν να γράψουν οι εφαρμογές. Οι ίδιες οι υλοποιήσεις μπορούν στη συνέχεια να γραφτούν από ανεξάρτητους τρίτους προμηθευτές και να συνδεθούν απρόσκοπτα όπως απαιτείται. Επομένως, οι προγραμματιστές εφαρμογών μπορούν να επωφεληθούν από οποιονδήποτε αριθμό εφαρμογών που βασίζονται σε πάροχους χωρίς να χρειάζεται να

προσθέσουν ή να ξαναγράψουν κώδικα. Όπως προανέφερα τα κλειδιά ελέγχονται και ανανεώνονται κάθε 30 μέρες για επιπλέον ασφάλεια και ακεραιότητα των κλειδιών αυτών.

Τιμές κατακερματισμού

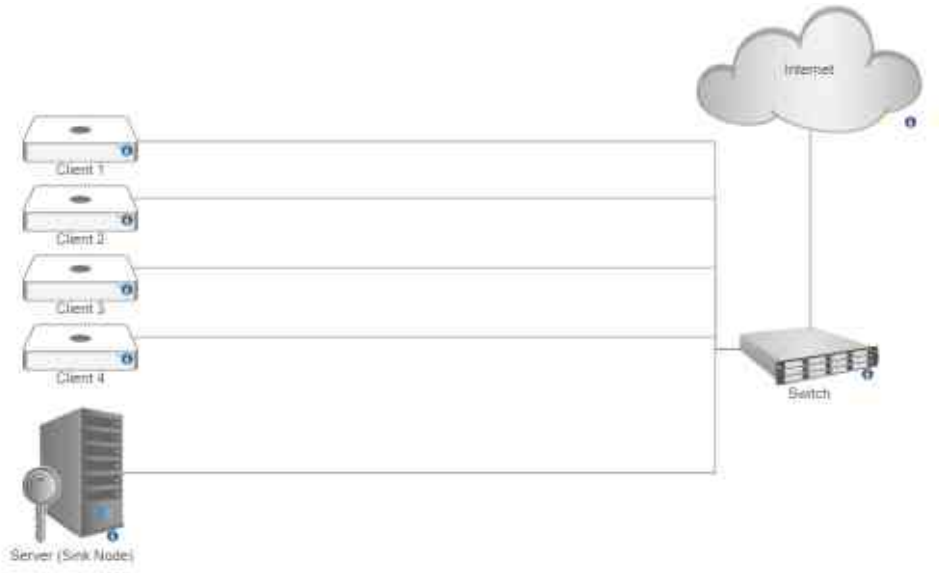
Οι τιμές κατακερματισμού δεν φυλάσσονται κάπου καθώς ο χρόνος υπολογισμού τους είναι μηδαμινός και με το να φυλάσσεται κάπου είναι μια αρκετά μεγάλη «τρύπα» ασφαλείας.

4.6.2 Σύνδεση τιμής κατακερματισμού πελάτη με μπλοκ πελάτη.

Για τη σύνδεση της τιμής κατακερματισμού του πελάτη με το μπλοκ του στην αλυσίδα, χρησιμοποιήθηκε η δομή δεδομένων “hashmap" της java, για το λόγω ότι προσφέρει την γρηγορότερη αναζήτηση αντικειμένων εντός της δομής. Η κλάση Java HashMap υλοποιεί τη διεπαφή χάρτη που μας επιτρέπει να αποθηκεύουμε ζεύγος κλειδιών και τιμών, όπου τα κλειδιά πρέπει να είναι μοναδικά. Εάν προσπαθήσετε να εισαγάγετε το διπλότυπο κλειδί, θα αντικαταστήσει το στοιχείο του αντίστοιχου κλειδιού. Είναι εύκολο να εκτελέσετε λειτουργίες χρησιμοποιώντας το ευρετήριο κλειδιού όπως ενημέρωση, διαγραφή κ.λπ. Η κλάση HashMap βρίσκεται στο πακέτο java.util. Επιπλέον, όπως δηλώνει και το όνομα η δομή αυτή είναι η καταλληλότερη για τιμές κατακερματισμού.

4.7 Τοπολογία Δικτύου

Όπως προανέφερα για το δίκτυο χρησιμοποιήθηκαν 5 Raspberry Pie3 τα οποία 4 θα έχουν το ρόλο των client και 1 θα έχει το ρόλο του server(Sink node).Για την συγκεκριμένη εργασία χρησιμοποιήθηκε η τοπολογία Sink in the Middle παρόλο που το πρωτόκολλο δεν περιορίζεται από καμία τοπολογία (μπορεί να εφαρμοστεί σε όλες τις τοπολογίες η οποίες περιέχουν sink node). Η τοπολογία του δικτύου φαίνεται στο πιο κάτω διάγραμμα δικτύου. Αναπαράσταση της τοπολογίας του δικτύου φαίνεται στο πιο κάτω Σχήμα 13.



Σχήμα 13

4.8 Λειτουργικό σύστημα μηχανών

Για τις 5 μηχανές Raspberry Pie3 χρησιμοποιήθηκε το λειτουργικό σύστημα Raspberry Pi OS (64-bit) με τη χρήση του εργαλείου Raspberry PI imager. Το Raspberry Pi OS είναι ένα λειτουργικό σύστημα που μοιάζει με Unix και βασίζεται στη διανομή Debian Linux για την οικογένεια συμπαγών υπολογιστών με μία πλακέτα Raspberry Pi. Αναπτύχθηκε για πρώτη φορά ανεξάρτητα το 2012, και έχει παραχθεί ως το κύριο λειτουργικό σύστημα για αυτές τις πλακέτες από το 2013, που διανέμεται από το Ίδρυμα Raspberry Pi. Αφού βασίζεται σε διανομή Debian Linux, είναι το καταλληλότερο για να τρέξει το java πρόγραμμα μας.

Κεφάλαιο 5
Αξιολόγηση απόδοσης υλοποίησης μέσω πειραμάτων

5.1 Επισκόπηση

Σε αυτό το κεφάλαιο θα δούμε τα διάφορα προβλήματα που προέκυψαν στην πορεία της παρούσας διπλωματικής εργασίας καθώς και πως διεξάχθηκαν πειράματα για μέτρηση της απόδοσης και της ασφάλειας του πρωτοκόλλου.

5.2 Προβλήματα που προέκυψαν.

5.2.1 Λογισμικό για Raspberry Pi

Στην αρχή της παρούσας διπλωματικής εργασίας επιλέχθηκε το λογισμικό Raspberry Pi OS (32-bit), το οποίο λογισμικό παρουσίασε διάφορα προβλήματα με την java 17 η οποία χρησιμοποιήθηκε για να τρέξουν τα διάφορα jar αρχεία για την εκτέλεση των πειραμάτων. Έτσι, μετά από διάφορες αναταραχές που προκλήθηκαν με την έκδοση αυτή αποφασίσαμε να χρησιμοποιήσουμε το Raspberry PI OS (64-bit), το οποίο όχι μόνο υποστήριζε καλύτερα την συγκεκριμένη έκδοση της java αλλά επιπλέον παρείχε μια πιο ομαλή εμπειρία.

5.2.2 Συγχρονισμός ζώνης ώρας μηχανών.

Στην παρούσα διπλωματική εργασία για λόγους ασφαλείας όπως προανάφερα και πιο πάνω χρησιμοποιήθηκαν χρονικές σφραγίδες για όλα τα πακέτα που ανταλλάσσονται μεταξύ διακομιστή και πελάτη. Στα πειράματα μας παρατηρήθηκαν μεγάλη και τυχαία

διαφορά μεταξύ των ωρών στις διάφορες μηχανές με αποτέλεσμα όλα τα πακέτα λόγω μεγάλης διαφοράς στις χρονικές σφραγίδες απορρίπτονταν. Έτσι μετά από έρευνα διαπιστώθηκε πως στο λογισμικό παρόλο που έβαζες σε όλες τις μηχανές την ίδια ζώνη ώρας (π.χ. ώρα Ελλάδας) οι ώρες μεταξύ των μηχανών διέφεραν. Έτσι για να λυθεί το πρόβλημα αυτό χρειάστηκε να χρησιμοποιηθεί το πρωτόκολλο NTP.

Το πρωτόκολλο ώρας δικτύου (NTP) είναι ένα πρωτόκολλο Διαδικτύου που χρησιμοποιείται για συγχρονισμό με πηγές ώρας ρολογιού υπολογιστή σε ένα δίκτυο. Ανήκει και είναι ένα από τα παλαιότερα μέρη της σουίτας TCP/IP. Ο όρος NTP ισχύει τόσο για το πρωτόκολλο όσο και για τα προγράμματα πελάτη-διακομιστή που εκτελούνται σε υπολογιστές.

Για να συγχρονιστούν οι ώρες των μηχανών χρειάστηκε να συγχρονιστεί ο να κατεβάσουμε την υπηρεσία ntp στον διακοσμητή μας και στη συνέχεια να τροποποιήσουμε το αρχείο /etc/ntp.conf έτσι ώστε να περιέχει του διάφορους διακοσμητές που θα μας παρείχαν ώρα. Στη συνέχεια, επανεκκινούμε την υπηρεσία ntp με την εντολή “sudo service ntp restart”. Επιπρόσθετα, κατεβάζουμε την υπηρεσία ntpdate στους πελάτες μας και στο αρχείο /etc/hosts των πελατών εισάγουμε τον διακομιστή μας με την ip του για να είναι γνωστή και από τις υπόλοιπες μηχανές. Επιπλέον, κατεβάζουμε την υπηρεσία ntp στους πελάτες μας και τροποποιούμε το αρχείο /etc/ntp.conf βάζοντας σαν διακομιστή τον διακομιστή μας. Τέλος, συγχρονίζουμε τα ρολόγια των πελατών με την εντολή “sudo ntpdate server”, όπου server είναι το όνομα που δώσαμε στον διακομιστή μας.

Μετά τη χρήση του πρωτοκόλλου αυτού, όλες οι μηχανές είχαν την ίδια ώρα και έτσι το πείραμα μπόρεσε να συνεχιστεί κανονικά.

5.2.3 Φθορά καρτών SD

Μετά από πολλούς πειραματισμούς και εγγραφές εικόνας (λογισμικού) πάνω στις κάρτες SD παρατηρήθηκε πως κάποιες από τις κάρτες αυτές φθάρηκαν σε σημείο που δεν μπορούσαν να χρησιμοποιηθούν. Έτσι, αναγκαστήκαμε να προμηθευτούμε περισσότερες κάρτες SD.

5.2.4 Έκδοση Java.

Όπως προανέφερα για την παρούσα διπλωματική εργασία χρησιμοποιήθηκε java 17 για να κτιστούν τα jar αρχεία (clients.jar και server.jar) που εκτελούνται από τις μηχανές, έτσι μόνο με την συγκεκριμένη έκδοση ή με μια νέα έκδοση μπορούσαν να

εκτελεστούν τα αρχεία αυτά. Στην αρχή της πειραματικής φάσης η java εγκαταστάθηκε στις μηχανές χρησιμοποιώντας την εντολή sudo apt install default-jdk η οποία εγκαταστούσε το 8ο jdk της java το οποίο δεν μπορούσε να χρησιμοποιηθεί για τα πειράματα μας. Έτσι, έπρεπε να χρησιμοποιηθεί η εντολή sudo apt install openjdk-17-jdk -y έτσι ώστε να εγκατασταθεί η 17η έκδοση της java με την οποία μπορούσαμε να εκτελέσουμε τα αρχεία jar στις μηχανές μας.

5.2.5 Διάδοση Εκπομπής

Στην πρώτη προσέγγιση του 2ου πειράματος είχαμε 4 μηχανές και ένα υπολογιστή με λειτουργικό Windows 10 ενωμένα πάνω σε ένα switch 5 θυρών με απόδοση 100 Mbps (half duplex) και 200Mbps (full duplex). Στις μηχανές δόθηκαν καθώς και στον υπολογιστή δόθηκαν στατικές διευθύνσεις Ip για να είναι όλες εντός του ίδιου δικτύου. Το πρόβλημα με αυτή την προσέγγιση είναι ότι λόγω τις φύσης του switch δεν εμφανίζονταν τα πακέτα που ανταλλάσσονταν από τα Raspberry Pi στο wireshark του υπολογιστή με τα Windows 10. Έτσι, έπρεπε να στηθεί ένα Access Point σε ένα Raspberry Pi, έτσι ώστε να δίνει εκείνο τις διευθύνσεις και τα πακέτα να μπορούν να γίνονται broadcast σε όλο το δίκτυο. Για τον τρόπο με τον οποίο στήθηκε το Access Point θα μιλήσουμε στο παράρτημα

5.3 Πειράματα

Στην παρούσα διπλωματική εργασία διεξάχθηκαν 2 πειράματα. Το 1ο πείραμα έγινε τοπικά, δηλαδή οι μηχανές μας είχαν στατικές ιδιωτικές διευθύνσεις IP με κύριο σκοπό την μέτρηση τις επίδοσης και των πόρων που χρειάζεται το πρωτόκολλο. Το 2ο πείραμα χρησιμοποιήθηκαν δυναμικές ιδιωτικές διευθύνσεις για τους 4 πελάτες και στατική ιδιωτική διεύθυνση IP για τον διακομιστή. Το πείραμα αυτό είχε ως σκοπό όχι μόνο την μέτρηση της επίδοσης του πρωτοκόλλου σε ένα πιο πιθανό σενάριο αλλά και την αξιολόγηση του πρωτοκόλλου σε διάφορες κοινότυπες επιθέσεις οι οποίες αναφέρθηκαν προηγουμένως. Οι επιθέσεις διεξάχθηκαν από υπολογιστή ο οποίος όχι μόνο είχε πρόσβαση στο δίκτυο του πειράματος , αλλά χρησιμοποιούσε και το εργαλείο Kali Linux. Το Kali Linux είναι μια διανομή Linux ανοιχτού κώδικα, βασισμένη στο Debian και προσανατολισμένη σε διάφορες εργασίες ασφάλειας πληροφοριών, όπως Δοκιμή διείσδυσης, Έρευνα ασφαλείας, Εγκληματολογία Υπολογιστών και Αντίστροφη Μηχανική. Για την εκτέλεση του πρωτοκόλλου χρειάστηκαν δύο προγράμματα java,

ένα για τους πελάτες και ένα για τον διακομιστή. Μετά την δημιουργία των δύο αυτών προγραμμάτων χρειάστηκε να δημιουργηθούν δύο εκτελέσιμα αρχεία με τη βοήθεια του εργαλείου NetBeans. Για τα προγράμματα χρησιμοποιήθηκε το jdk 17, ένα από τα νεότερα κιτ ανάπτυξης της γλώσσας προγραμματισμού java.

5.4.1 Τοπικό πείραμα

Για το πείραμα αυτό οι 5 μηχανές μας συνδέθηκαν πάνω σε ένα switch 5 θυρών με απόδοση 100 Mbps (half duplex) και 200Mbps (full duplex). Στη συνέχεια δόθηκαν στατικές IP διευθύνσεις στο δίκτυο 192.168.10.1 – 192.168.10.6 . Στη μηχανή με διεύθυνση 192.168.10.1 δόθηκε ο ρόλος του διακομιστή ενώ στις υπόλοιπες 4 μηχανές ο ρόλος των πελατών. Σκοπός του πειράματος αυτού ήταν η μέτρηση της επίδοσης του πρωτοκόλλου ως προς τον χρόνο εκτέλεσης (δηλαδή ο χρόνος που χρειάζεται για να καταχωρήσει και να ταυτοποίησή μια μηχανή) και ως προς τους πόρους ππου χρειάζεται το πρωτόκολλο, όπως μνήμη, επεξεργαστή και ενέργεια καθώς οι πόροι αυτοί είναι ζωτικής σημασίας για τις μηχανές μας.

5.4.2 Λειτουργία δρομολόγησης διακομιστή

Για να λειτουργήσει ως δρομολογητής ο διακομιστής του πειράματος μας έπρεπε να τροποποιηθεί το αρχείο /etc/dhcpcd.conf το οποίο αρχείο μας επιτρέπει να βάλουμε στατικές διευθύνσεις στις μηχανές μας. Για τους πελάτες χρησιμοποιήθηκαν οι διευθύνσεις 192.168.1.2 – 192.168.1.5 και σαν gateway η διεύθυνση του διακομιστή μας (192.168.1.1). Ενώ για τον διακομιστή χρησιμοποιήθηκε η διεύθυνση 192.168.1.1 και σαν gateway βάλαμε “nogateway” διότι θέλαμε να περνούν τα πακέτα από το Wlan το οποίο είναι το δίκτυο του σπιτιού μου. Έτσι καταφέραμε όλες οι μηχανές να μπορούν να επικοινωνούν μεταξύ τους και ο διακοσμητής μας να έχει πρόσβαση στο διαδίκτυο.

Το αρχείο /etc/dhcpcd.conf για τον πελάτη και τον διακομιστή φαίνεται στο πιο κάτω Σχήματα 14 1 και 14 2 ανάλογα.

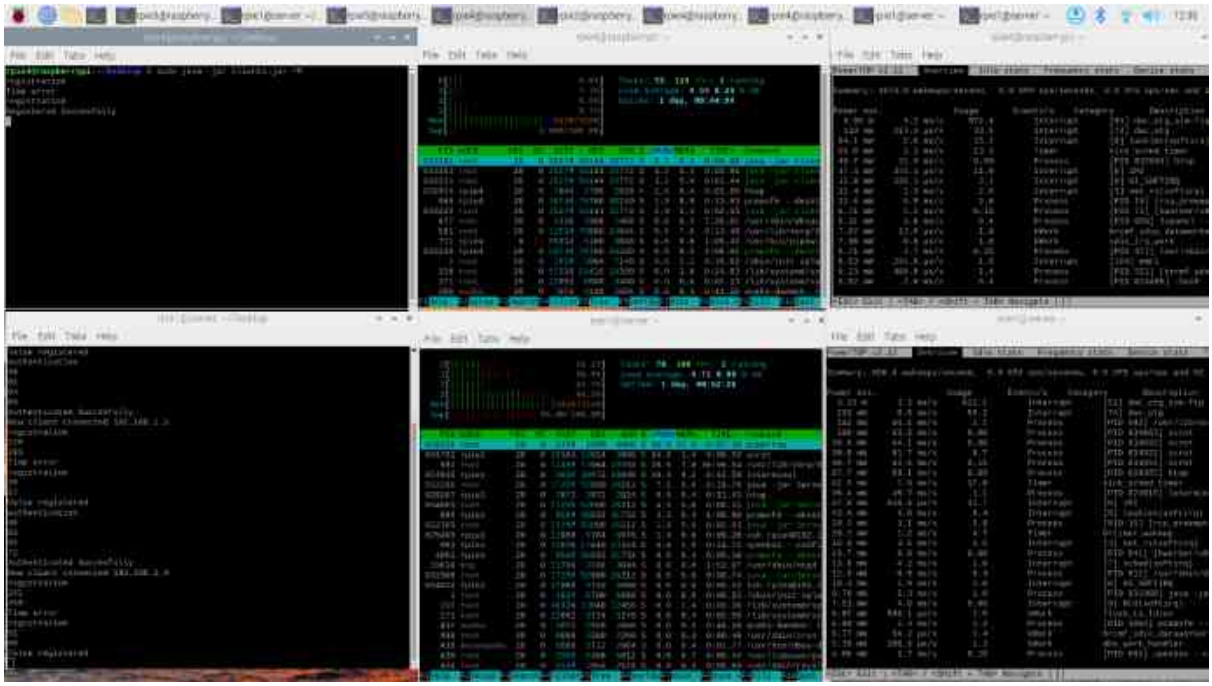
Σχήμα 14 1

Σχήμα 14 2

5.5 Αποτελέσματα τοπικού πειράματος

Για να πάρουμε αποτελέσματα όσο αφορά το πείραμα μας χρησιμοποιήθηκαν οι εντολές htop για τους πόρους που χρειάζεται το πρωτόκολλο μας καθώς και η εντολή powertop για την ενέργεια που χρειάζεται το πρωτόκολλο μας. Στο πείραμα αυτό, οι πελάτες συνδέονταν ένας ένας με τον διακομιστή και πραγματοποιούσαν την πρώτη

τους δημιουργία κλειδιών, πακέτων εγγραφής και εγγραφή στο blockchain του διακομιστή, για αυτό το λόγο γίνεται μεγάλη χρήση πόρων και ενέργειας κατά την πρώτη σύνδεση με τον διακομιστή. Επιπλέον, πρέπει να σημειωθεί πως καταγράφηκαν ένδειξης σε τρεις φάσεις του πειράματος. 1. Δημιουργία απαραίτητων αρχείων όπως πακέτα και κλειδιά και πρώτη σύνδεση με διακομιστή. 2. Πρώτη εγγραφή στο blockchain του διακομιστή. 3. Πρώτη αυθεντικοποίηση από τον διακομιστή. Αξιοσημείωτο το γεγονός ότι το πείραμα έγινε και με τους 4 πελάτες να στέλνουν πακέτα στον διακομιστή παράλληλα αφού ο διακομιστής μας είναι πολυνηματικός. Τέλος, σε κάθε μία από τις πιο πάνω φάσεις καταγράφηκαν μετρήσεις τόσο από τους πελάτες αλλά και από τον ίδιο τον διακομιστή. Ο τρόπος συλλογής πληροφοριών φαίνεται στο πιο κάτω Σχήμα 15.



Σχήμα 15 1

5.5.1 Συμπεράσματα αποτελεσμάτων Τοπικού Πειράματος

Σε αυτό το παράρτημα θα σχολιάσουμε τις ενδείξεις που πήραμε από τις τρεις φάσεις του πειράματος μας. Επιπλέον, πρέπει να αναφέρω ότι οι τιμές στις γραφικές παραστάσεις έγιναν λογαριθμικά με βάση το 10 για να είναι πιο ευδιάκριτες, έτσι οι τιμές που είναι μικρότερες από 1 βρίσκονται στην κάτω μεριά της γραφικής.

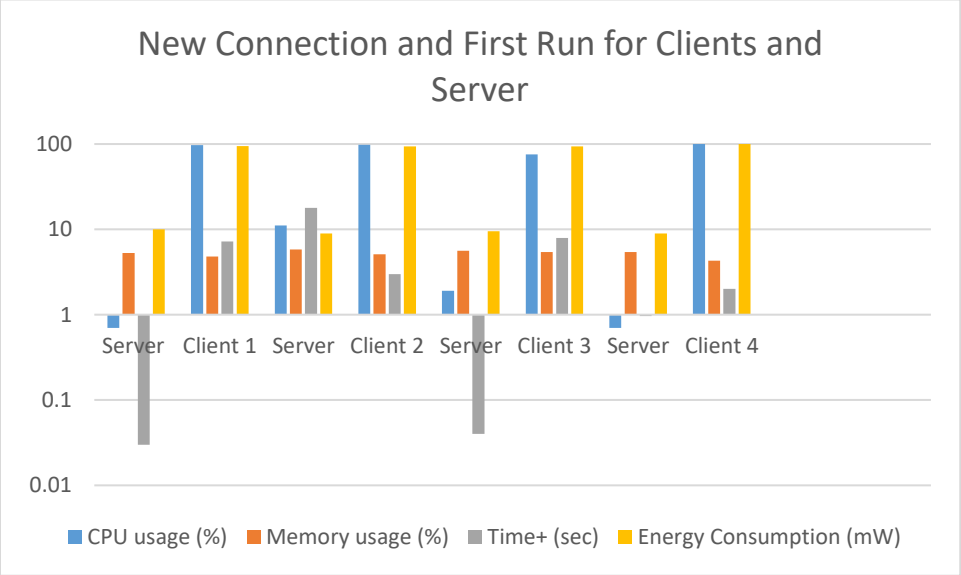
5.5.2 Συμπεράσματα αποτελεσμάτων Πρώτης Σύνδεσης

5.5.2.1 Πελάτες

Από την γραφική παράσταση παρατηρούμε ότι έχουμε πολύ μικρή κατανάλωση ενέργειας και μεγάλη χρήση του επεξεργαστή. Αυτό, είναι λογικό και επόμενο αφού το πρόγραμμα μας τρέχει για πρώτη φορά, που σημαίνει πως πρέπει να δημιουργήσει όλα τα απαραίτητα αρχεία και κλειδιά που χρειάζεται το οποίο φυσικά δεν χρειάζεται και ιδιαίτερο χρόνο έτσι δεν καταναλώνεται πολύ ενέργεια. Επιπλέον, παρατηρούμε ότι η μνήμη που χρησιμοποιείται είναι ελάχιστη, πράγμα που είναι πολύ καλό αφού οι μηχανές μας δεν έχουν και ιδιαίτερα μεγάλη μνήμη.

5.5.2.2 Διακομιστής

Από την γραφική παράσταση παρατηρούμε ότι έχουμε πολύ μικρή κατανάλωση ενέργειας και ελάχιστη χρήση πόρων. Αυτό, είναι λογικό και επόμενο αφού ο διακομιστή δεν χρειάζεται επιπλέον αρχεία εκτός από τα κλειδιά του και χάρη στη πολυνηματική του φύση δεν παρατηρούμε μεγάλες αυξήσεις στην χρήση πόρων με κάθε νέα σύνδεση.



Γραφική 1

New Connection								
	Server	Client 1	Server	Client 2	Server	Client 3	Server	Client 4
CPU usage (%)	0.7	97.4	11.1	98.2	1.9	75.4	0.7	100
Memory usage (%)	5.3	4.8	5.8	5.1	5.6	5.4	5.4	4.3
Time+ (sec)	0.03	7.19	17.91	2.98	0.04	7.94	0.96	2.01
Energy Consumption (mW)	10	94.3	8.94	94.2	9.53	94.2	8.93	100

Δεδομένα Γραφικής 1

5.5.3 Συμπεράσματα αποτελεσμάτων Εγγραφής

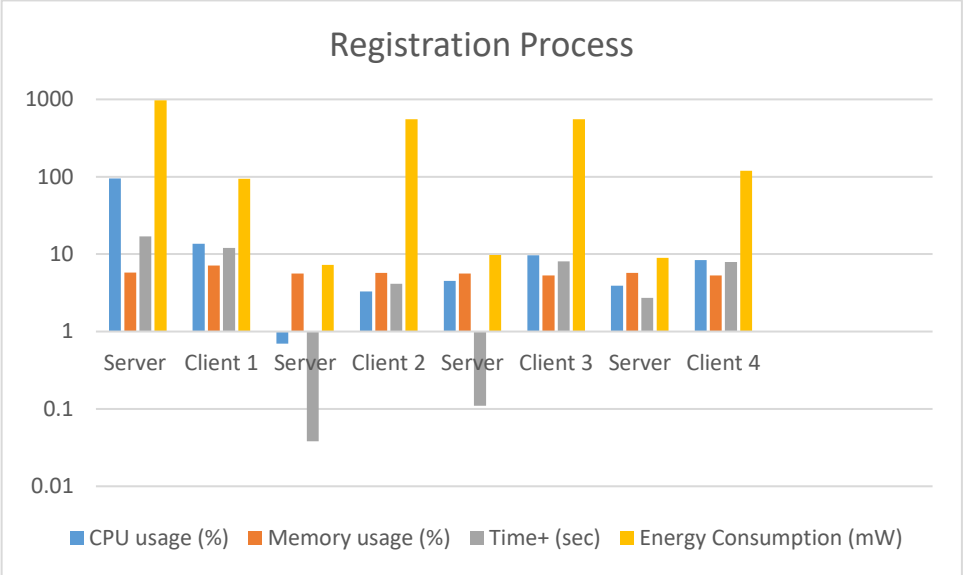
5.5.3.1 Πελάτες

Από την γραφική παράσταση παρατηρούμε ότι έχει απαιτείται περισσότερη ενέργεια και αυτό οφείλεται στην κρυπτογραφία που απαιτείται καθώς και στις χρονικές σφραγίδες που χρειάζονται για την ασφαλή αποστολή πακέτων στον διακομιστή. Από την άλλη, βλέπουμε πως η χρήση του επεξεργαστή μειώνεται δραματικά και αυτό οφείλεται στο ότι δεν χρειάζεται μεγάλη υπολογιστή δύναμη αφού ότι αρχεία και κλειδιά χρειάζεται τα έχει ήδη δημιουργήσει. Τέλος, ο χρόνος και η μνήμη που απαιτείται έχει αυξηθεί ελάχιστα αφού χρειάζεται να φέρει από την μνήμη τα απαραίτητα κλειδιά και πακέτα που χρειάζεται για την εγγραφή και αφού περιμένει απάντησή από τον διακομιστή λογικό είναι ο χρόνος εκτέλεσης να αυξηθεί ελάχιστα.

5.5.3.2 Διακομιστής

Από την γραφική παράσταση παρατηρούμε ότι έχουμε σημαντική αύξηση στην κατανάλωση ενέργειας πράγμα που δεν είναι ανησυχητικό καθώς μετά από έρευνα η μέγιστη τιμή που καταναλώνεται (972 mW) είναι μηδαμινή μπροστά στην ενέργεια που χρειάζεται ένα απλό πληκτρολόγιο (Σχήμα 16). Επιπλέον βλέπουμε σημαντική αύξηση στη χρήση του επεξεργαστή το οποίο αναμένετέ διότι χρειάζεται να κάνει κάποιους υπολογισμούς σε σχέση με τις χρονικές σφραγίδες των πακέτων να αποκρυπτογραφήσει τα πακέτα και να ελέγξει το blockchain και τέλος να εγγράψει τον νέο πελάτη στο

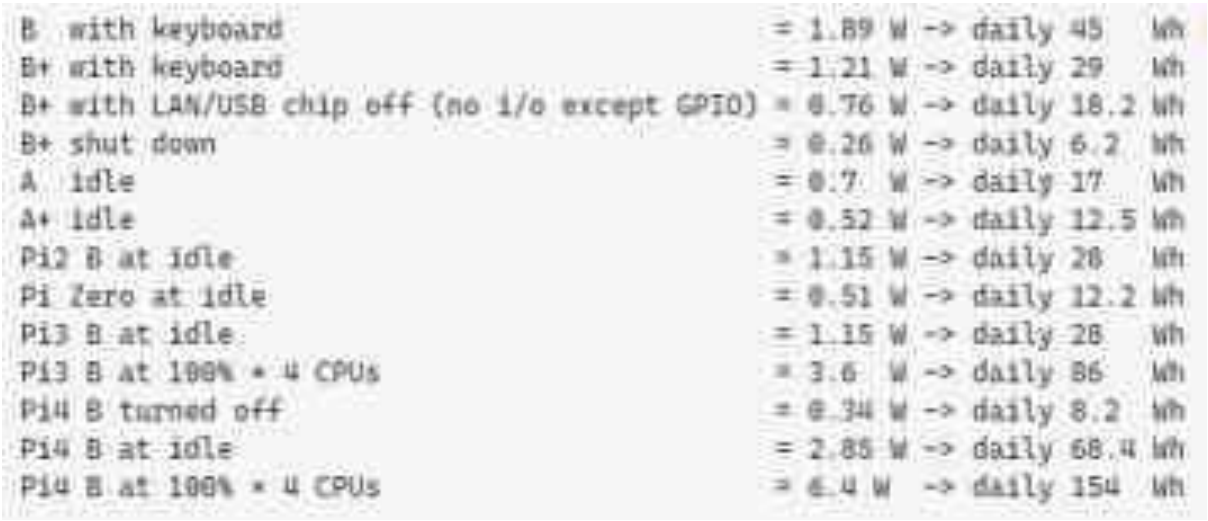
blockchain, ωστόσο δεν είναι μεγάλο ποσοστό. Επιπρόσθετα, η μνήμη καθώς και ο χρόνος εκτέλεσης παραμένει ο ίδιος.



Γραφική 2

Registration								
	Server	Client 1	Server	Client 2	Server	Client 3	Server	Client 4
CPU usage (%)	95.3	13.6	0.7	3.3	4.5	9.7	3.9	8.4
Memory usage (%)	5.8	7.1	5.6	5.7	5.6	5.3	5.7	5.3
Time+ (sec)	17.01	12.05	0.038	4.16	0.11	8.09	2.72	7.94
Energy Consumption (mW)	972	94.3	7.25	554	9.78	554	8.93	120

Δεδομένα Γραφικής 2



Σχήμα 16

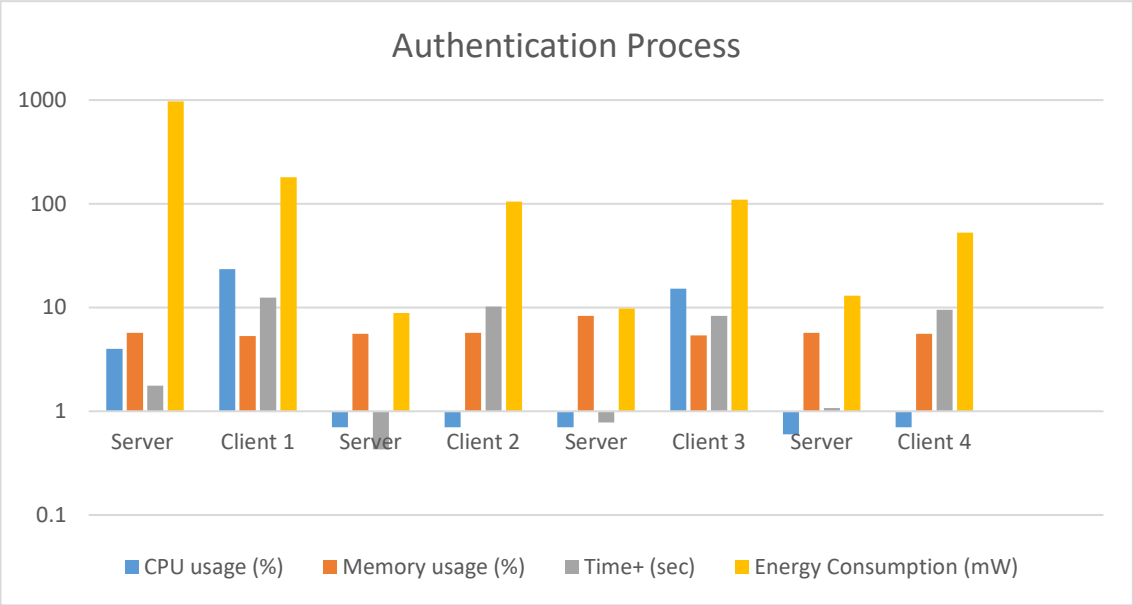
5.5.4 Συμπεράσματα αποτελεσμάτων Αυθεντικοποίησης

5.5.4.1 Πελάτες

Από την γραφική παράσταση παρατηρούμε ότι η κατανάλωση ενέργειας καθώς και οι πόροι που χρειάζεται το πρωτόκολλο μας έχει παραμείνει στα ίδια μήκη κύματος με την φάση της εγγραφής, βλέποντας ελάχιστη μείωση στην χρήση του επεξεργαστή.

5.5.4.2 Διακομιστής

Από την γραφική παράσταση παρατηρούμε ότι έχουμε μια μείωση στην κατανάλωση ενέργειας καθώς και σημαντική μείωση στην χρήση του επεξεργαστή. Ο χρόνος εκτέλεσης καθώς και η χρήση μνήμης παραμένει στα ίδια μήκη κύματος με την φάση της εγγραφής



Γραφική 3

Authentication								
	Server	Client 1	Server	Client 2	Server	Client 3	Server	Client 4
CPU usage (%)	4	23.5	0.7	0.7	0.7	15.2	0.6	0.7
Memory usage (%)	5.7	5.3	5.6	5.7	8.34	5.4	5.7	5.6
Time+ (sec)	1.76	12.42	0.43	10.19	0.78	8.34	1.07	9.51
Energy Consumption (mW)	972	180	8.87	105	9.78	110	13	52.9

Δεδομένα Γραφικής 3

5.5.5 Συμπεράσματα

Μετά από το πείραμα μας μπορούμε να πούμε με σιγουριά ότι το πρωτόκολλο μας δεν χρειάζεται ούτε πολύ ενέργεια ούτε πολλούς πόρους για να λειτουργήσει.

5.6.1 Πείραμα αληθινού σεναρίου

Για το πείραμα αυτό έπρεπε η μία από τις 5 μηχανές Raspberry Pi (η μηχανή που λειτουργεί και ως διακομιστής) να στηθεί ως Access Point έτσι ώστε να ενωθούν οι 5 μηχανές Raspberry Pi καθώς και ο υπολογιστής με λειτουργικό σύστημα Windows 10 σε ένα ενιαίο δίκτυο. Αφού όλες οι μηχανές ενωθούν πάνω στο δίκτυο αυτό μπορούμε να διεξάγουμε τις επιθέσεις που προαναφέρθηκαν στο παράρτημα 3 για να δούμε πως

ανταπεξέλθετε το πρωτόκολλο μας στις επιθέσεις αυτές. Για τις διάφορες επιθέσεις χρειάστηκε να στηθεί μια εικονική μηχανή με λογισμικό Kali Linux το οποίο μας παρέχει τα κατάλληλα εργαλεία για να διεξάγουμε τις επιθέσεις αυτές.

5.6.1.1 Εικονική Μηχανή

Μια εικονική μηχανή (VM) είναι μια ψηφιακή έκδοση ενός φυσικού υπολογιστή. Το λογισμικό εικονικής μηχανής μπορεί να εκτελεί προγράμματα και λειτουργικά συστήματα, να αποθηκεύει δεδομένα, να συνδέεται σε δίκτυα και να κάνει άλλες υπολογιστικές λειτουργίες και απαιτεί συντήρηση, όπως ενημερώσεις και παρακολούθηση συστήματος. Οι εικονικές μηχανές τρέχουν σε μια φυσική μηχανή και έχουν πρόσβαση σε υπολογιστικούς πόρους από λογισμικό που ονομάζεται hypervisor. Ο hypervisor αφαιρεί τους πόρους της φυσικής μηχανής σε ένα pool που μπορεί να παρασχεθεί και να διανεμηθεί όπως απαιτείται, επιτρέποντας σε πολλαπλά VM να λειτουργούν σε ένα μόνο φυσικό μηχάνημα.

5.6.1.2 Λογισμικό Kali Linux

Το Kali Linux (παλαιότερα γνωστό ως BackTrack Linux) είναι μια διανομή Linux ανοιχτού κώδικα, βασισμένη στο Debian που στοχεύει σε προηγμένες δοκιμές διείσδυσης και έλεγχο ασφαλείας. Αυτό το επιτυγχάνει παρέχοντας κοινά εργαλεία, διαμορφώσεις και αυτοματισμούς που επιτρέπουν στον χρήστη να εστιάσει στην εργασία που πρέπει να ολοκληρωθεί και όχι στην περιβάλλουσα δραστηριότητα.

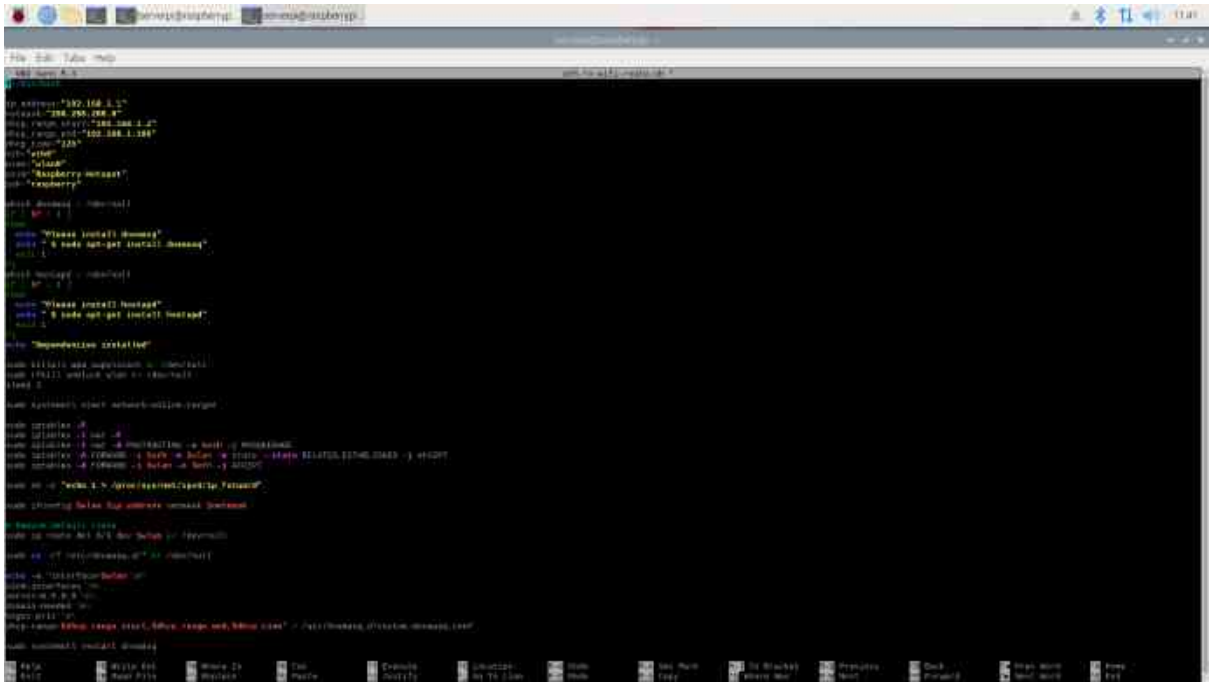
Το Kali Linux περιέχει συγκεκριμένες τροποποιήσεις του κλάδου καθώς και αρκετές εκατοντάδες εργαλεία που στοχεύουν σε διάφορες εργασίες Ασφάλειας Πληροφοριών, όπως Δοκιμή διείσδυσης, Έρευνα Ασφαλείας, Εγκληματολογία Υπολογιστών, Αντίστροφη Μηχανική, Διαχείριση ευπάθειας και Δοκιμή Red Team.

Το Kali Linux είναι μια λύση πολλαπλών πλατφορμών, προσβάσιμη και ελεύθερα διαθέσιμη σε επαγγελματίες ασφάλειας πληροφοριών και χομπίστες.

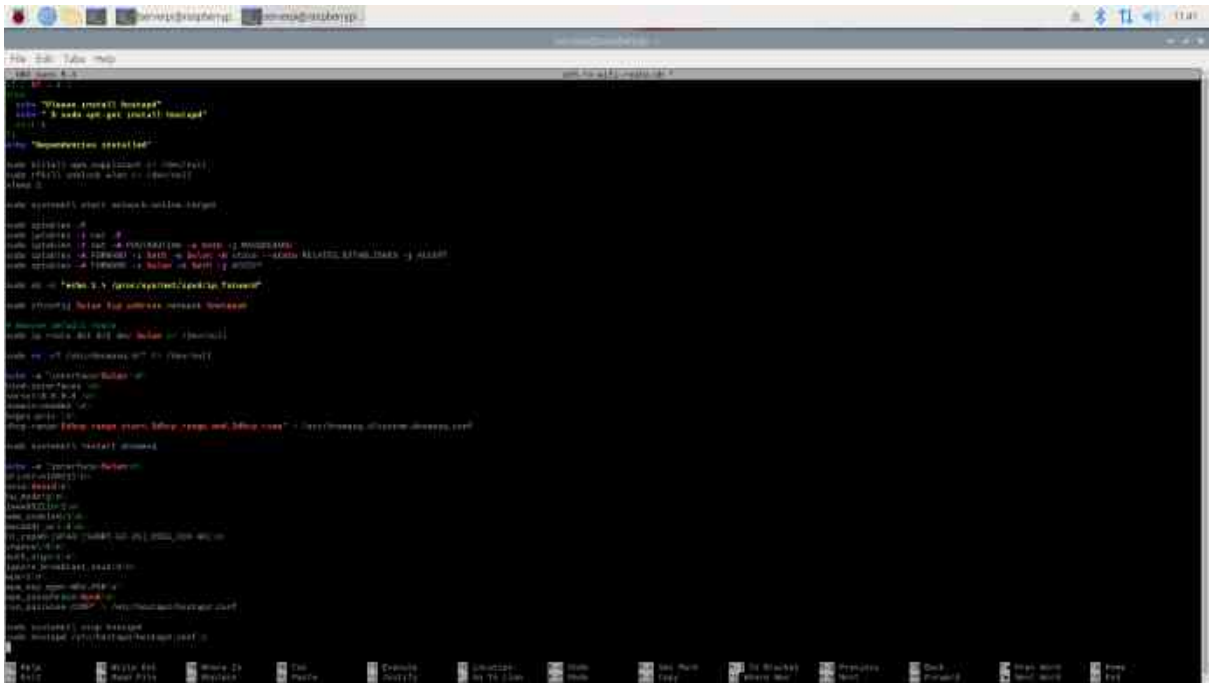
5.6.2 Λειτουργία δρομολόγησης διακομιστή

Για να λειτουργήσει ως Access Point η μηχανή Raspberry Pi χρησιμοποιήθηκαν τα εργαλεία DNSMASQ, HOSTAP και IPTABLES. Η μηχανή, είναι ενσύρματα ενωμένη στο δίκτυο του σπιτιού μου και βάση αυτού του δικτύου δίνει διευθύνσεις ασύρματα στις υπόλοιπες μηχανές. Για να πραγματοποιηθεί αυτή μας η ενέργεια χρειάστηκε να τρέξουμε ένα bash script (Σχήμα 17.1 και 17.2) το οποίο άλλαζε την διαμόρφωση των κατάλληλων αρχείων των εργαλείων που προανάφερα έτσι ώστε να δίνει πρόσβαση στο

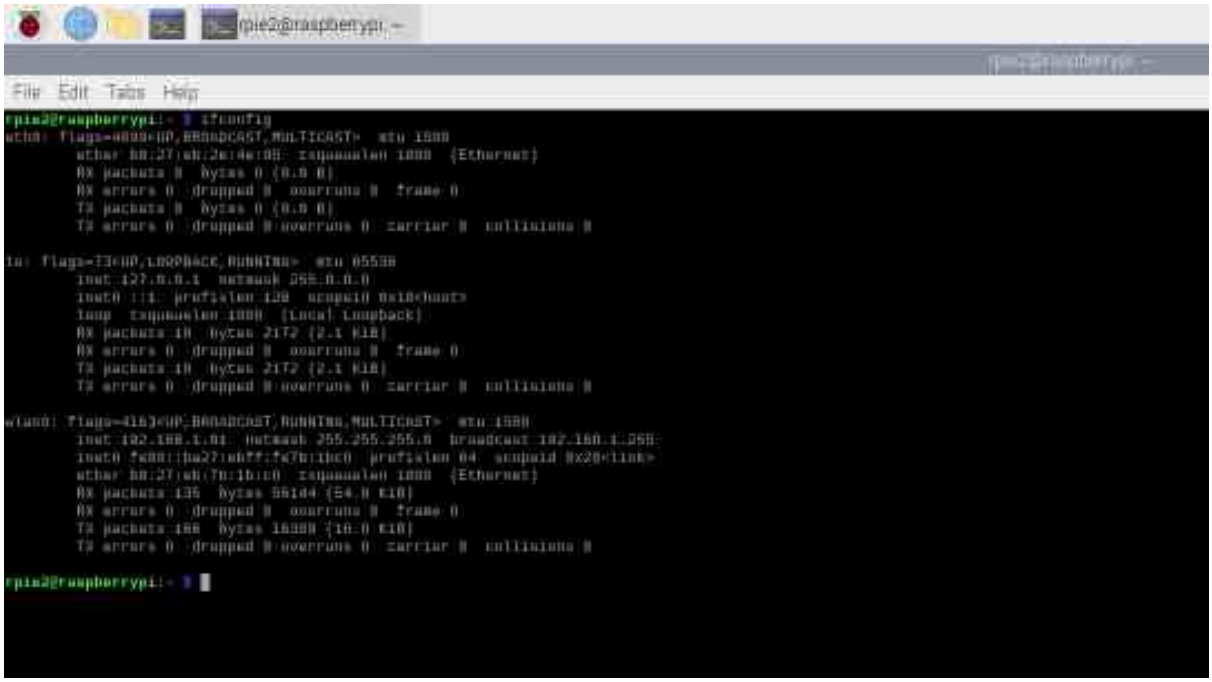
διαδίκτυο στις μηχανές σε ένα δικό τους ανεξάρτητο ασύρματο δίκτυο. Το script αυτό πρέπει να τρέξει μόλις ξεκινήσει η μηχανή μας, έτσι όλες οι μηχανές μπορούν να ενωθούν ασύρματα στο δίκτυο αυτό. Ο διακομιστής (που έχει και το ρόλο του δρομολογητή στο πείραμα αυτό) έχει διεύθυνση 192.168.0.20 στο ενσύρματο δίκτυο που είναι ενωμένος ενώ στο ασύρματο δίκτυο που παρέχει έχει σαν διεύθυνση το 192.168.1.1 και δίνει διευθύνσεις από το 192.168.1.2 – 192.168.1.100, στη δική μας περίπτωση έδωσε σε μία μηχανή την διεύθυνση 192.168.1.81 (Σχήμα 18). Το ασύρματο δίκτυο όπως φαίνεται και στο πιο κάτω σχήμα έχει σαν ονομασία το ¨Raspberry-Hotspot¨. Επιπλέον, μόλις τρέξουμε το script, μπορούμε να δούμε ποια μηχανή συνδέεται και αποσυνδέεται από το δίκτυο μας καθώς αν πάει κάτι λάθος με τα κλειδιά που πρέπει να ανταλλαχθούν (Σχήμα 19).



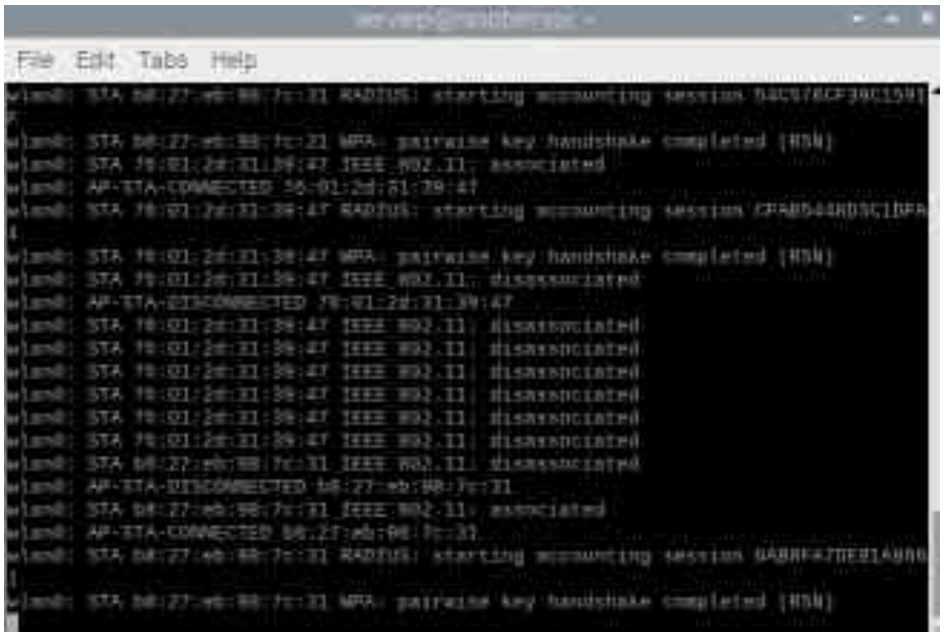
Σχήμα 17 1



Σχήμα 17 2



Σχήμα 18



Σχήμα 19

5.7 Αποτελέσματα πειράματος αληθινού σεναρίου

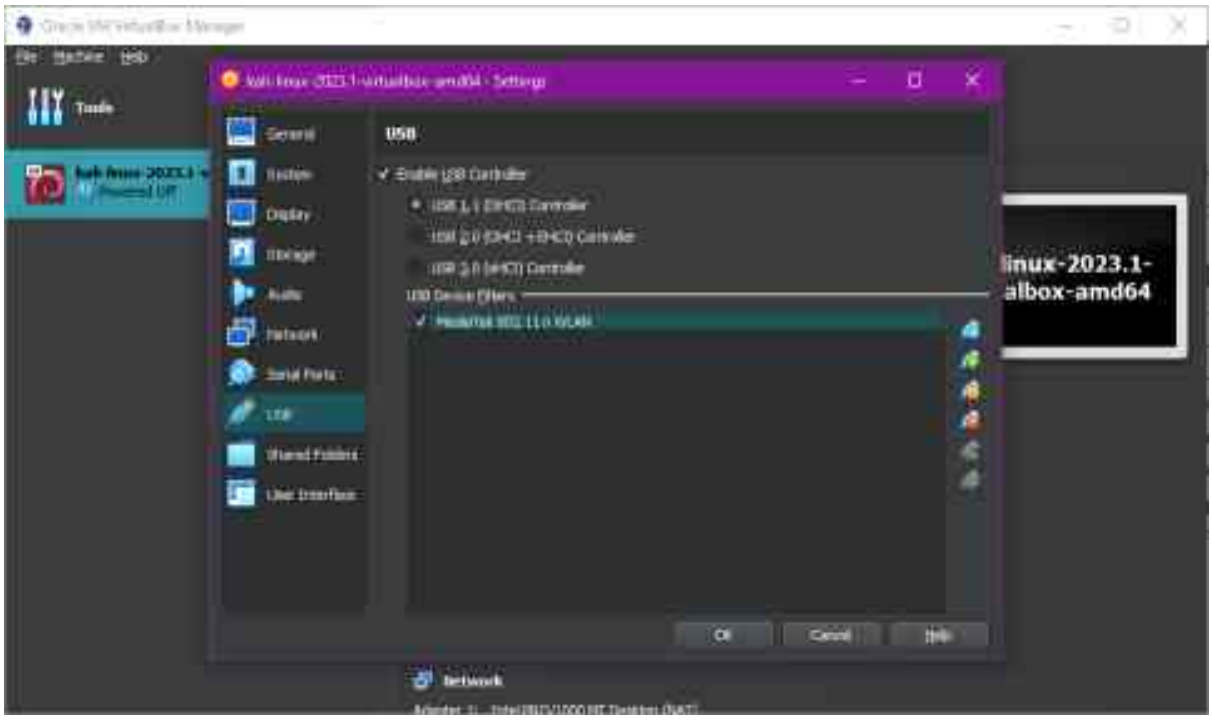
Στο παράρτημα αυτό θα δούμε πως αντεπεξέλθηκε το πρωτόκολλο μας στις διάφορες επιθέσεις που αναφερθήκαν πιο πάνω, χρησιμοποιώντας το λογισμικό Kali Linux.

5.7.1 Man-in-the-Middle

Για την επίθεση αυτή χρησιμοποιήθηκε το εργαλείο Wireshark και το εργαλείο Kismet στην εικονική μηχανή με λογισμικό Kali Linux. Χάρη στο εργαλείο Kismet καταφέραμε να καταγράψουμε τα πακέτα που ανταλλάσσονταν εντός του δικτύου, στη συνέχεια δημιουργεί ένα αρχείο τύπου .Kismet το οποίο μετατρέψαμε σε .pcapng (Σχήμα 20) για να μπορούμε να το φορτώσουμε στο εργαλείο Wireshark έτσι ώστε να μπορούμε να δούμε αναλυτικά το περιεχόμενο των διαφόρων πακέτων που καταγράφηκαν. Για να μπορέσουμε να έχουμε πρόσβαση στο δίκτυο των Raspberry Pi έπρεπε να συνδέσουμε μια κάρτα δικτύου στην εικονική μηχανή ως αφαιρούμενη συσκευή (Σχήμα 21). Έπειτα αφού συνδεθούμε στο δίκτυο “Raspberry-Hotspot” (Σχήμα 22) πρέπει να βάλουμε την κάρτα δικτύου σε λειτουργία παρακολούθησης με την εντολή ¨sudo iwconfig wlan0 mode monitor¨ (Σχήμα 23) όπου wlan0 το όνομα της κάρτας δικτύου, μπορούμε να ελέγξουμε να μπήκε η κάρτα μας σε λειτουργία παρακολούθησης με την εντολή ¨sudo iwconfig¨ (Σχήμα 23). Στη συνέχεια, αρχίσαμε την καταγραφή με την εντολή ¨kismet -c wlan0¨ (Σχήμα 24)και τέλος αλλάξαμε την

μορφή του αρχείου που αναπαράγει το εργαλείο Kismet σε pcapng για να μπορεί να φορτωθεί στο εργαλείο Wireshark. Στη συνέχεια, ανοίγουμε το αρχείο τύπου pcapng στο εργαλείο Wireshark, για να αναλύσουμε τα πακέτα που ανταλλάχτηκαν από τις μηχανές Raspberry Pi χρησιμοποιώντας το πρωτόκολλο που δημιούργησα έτσι ώστε να μπορούμε να επαληθεύσουμε ότι ένας επιτιθέμενος δεν μπορεί να δει τα δεδομένα των πακέτων. Βάλαμε σαν φίλτρο τις διευθύνσεις των δύο μηχανών μας και στη συνέχεια στείλαμε διάφορα πακέτα μεταξύ των μηχανών αυτών. Όπως είναι επόμενο τα πακέτα φαίνονταν μέσα στο εργαλείο, παρόλα αυτά όμως το περιεχόμενο τους είναι κρυπτογραφημένο (Σχήμα 25). Έτσι, ο επιτιθέμενος χωρίς να έχει τα κλειδιά κρυπτογράφησης (AES, RSA) δεν μπορεί να αποκρυπτογραφήσει τα περιεχόμενα των πακέτων αυτών, κάνοντας έτσι το πρωτόκολλο μας ανθεκτικό σε επιθέσεις Man-In-The-Middle.

Σχήμα 20

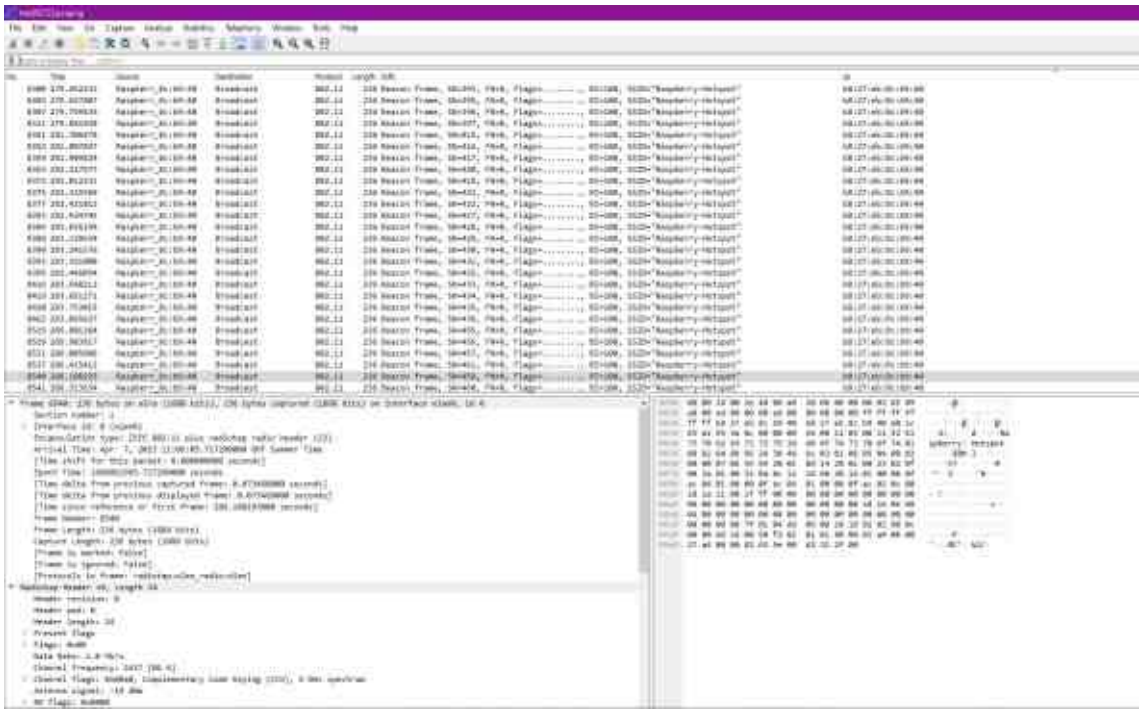


Σχήμα 21

Σχήμα 22

Σχήμα 23

Σχήμα 24



Σχήμα 25

5.7.1.1 Εργαλείο Wireshark

Το Wireshark είναι ένας αναλυτής πακέτων δικτύου. Ένας αναλυτής πακέτων δικτύου παρουσιάζει δεδομένα πακέτων που έχουν συλληφθεί με όσο το δυνατόν περισσότερες λεπτομέρειες.

Θα μπορούσατε να σκεφτείτε έναν αναλυτή πακέτων δικτύου ως μια συσκευή μέτρησης για την εξέταση του τι συμβαίνει μέσα σε ένα καλώδιο δικτύου, ακριβώς όπως ένας ηλεκτρολόγος χρησιμοποιεί ένα βολτόμετρο για να εξετάσει τι συμβαίνει μέσα σε ένα ηλεκτρικό καλώδιο (αλλά σε υψηλότερο επίπεδο, φυσικά).

Στο παρελθόν, τέτοια εργαλεία ήταν είτε πολύ ακριβά, είτε ιδιόκτητα είτε και τα δύο. Ωστόσο, με την εμφάνιση του Wireshark, αυτό άλλαξε. Το Wireshark είναι διαθέσιμο δωρεάν, είναι ανοιχτού κώδικα και είναι ένας από τους καλύτερους αναλυτές πακέτων που διατίθενται σήμερα.

5.7.1.2 Εργαλείο Kismet

Το kismet είναι ένα πλαίσιο ασύρματου δικτύου και συσκευής ανιχνευτή, sniffer, wardriving tool και πλαίσιο WIDS (wireless intrusion detection).

Το Kismet λειτουργεί με διεπαφές Wi-Fi, διεπαφές Bluetooth, κάποιο υλικό SDR (ραδιόφωνο που ορίζεται από λογισμικό) όπως το RTLSDR και άλλο εξειδικευμένο υλικό λήψης.

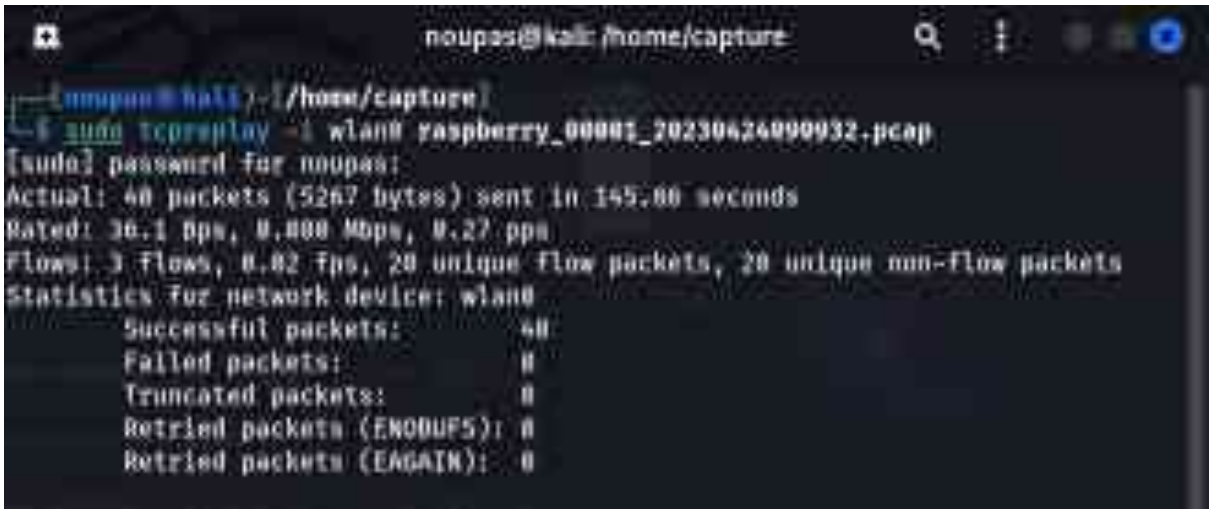
5.7.2 Replay Attack

Για την επίθεση αυτή χρησιμοποίησα ένα προ υπάρχον script (Σχήμα 26) που χρησιμοποίησε ο Δρ. Ιωάννου Ιάκωβος σε προηγούμενη του μελέτη το οποίο με την χρήση του εργαλείου Tshark και Cicflowmeter καταγράφει όλα τα πακέτα ενός δικτύου με την χρήση της κάρτας δικτύου και στη συνέχεια δημιουργεί ένα αρχείο τύπου pcap που θα το χρησιμοποιίσουμε στη συνέχεια για την επίθεση αυτή. Αφού τρέξουμε το script, αρχίζουμε να παρατηρούμε τα πακέτα που καταγράφονται (Σχήμα 27), παράλληλα στέλνουμε συνεχώς πακέτα εγγραφής και αυθεντικοποίησης μεταξύ πελάτη και διακομιστή για να βεβαιωθούμε τότι θα καταγράψουμε τα πακέτα που μας ενδιαφέρουν. Όποτε επιθυμούμε σταματάμε το script και μόλις σταματήσει η εκτέλεση του δημιουργεί ένα αρχείο pcap με την ονομασία που του θέσαμε μέσα στο script (στη

δική μας περίπτωση δημιούργησε το αρχείο raspberry.pcap) Στη συνέχεια, με την χρήση του εργαλείου TcpReplay στέλνουμε τα πακέτα πίσω στο δίκτυο (Σχήμα 28) και παρατηρούμε πως παρόλο που τα πακέτα στέλνονται, ο διακομιστής δεν τα δέχεται διότι δεν έγινε μέσω του πρωτοκόλλου μας. Αλλά στην περίπτωση που ο διακομιστής μας τα δεχόταν έχουμε χρησιμοποιήσεις για λόγους ασφαλείας όπως προανάφερα χρονικές σφραγίδες οι οποίες αν έχουν διαφορά μεγαλύτερη από 200 ms από την ώρα του διακομιστή δεν τα αποδέχεται. Έτσι, μπορούμε να πούμε με σιγουριά ότι το πρωτόκολλο μας είναι ασφαλής σε επιθέσεις tcpreplay.

Σχήμα 26

Σχήμα 27



Σχήμα 28

5.7.3 DDOS Attack

Όπως αναφέρθηκε και πιο πάνω μια επίθεση DDOS έχει ως σκοπό τη διακοπή των υπηρεσιών μίας εφαρμογής ή/και ενός ιστότοπου. Στην δική μας περίπτωση χρησιμοποιήσαμε το εργαλείο Goldeneye (εξηγήθηκε στο παράρτημα 3.4.2) για να διεξάγουμε μια τέτοια επίθεση. Από το σχήμα 26(Διακομιστής) και 27 (Πελάτης) μπορούμε να δούμε ότι το πρωτόκολλο μας χρησιμοποιεί την διεύθυνση του διακομιστή (192.168.1.1) καθώς και την πύλη 1235 για να μπορούν οι μηχανές να επικοινωνήσουν με τον διακομιστή. Έτσι, κάναμε μια επίθεση με στόχο την διεύθυνση και την πύλη αυτή. Για να χρησιμοποιήσουμε το εργαλείο Goldeneye χρησιμοποιήσαμε την εντολή “sudo goldeneye http://192.168.1.1:1235/ -s10 -m random” έτσι είχαμε 10 workers οι οποίοι είχαν 10 συνδέσεις ο καθένας πάνω στην πύλη που χρησιμοποιεί το πρωτόκολλο μας. Από ότι βλέπουμε και στο Σχήμα 28 καμία σύνδεση δεν επιτράπηκε καθώς το πρωτόκολλο μας δεν επιτρέπει συνδέσεις εκτός από αυτές που χρησιμοποιούν το πρωτόκολλο μας.

Σχήμα 26

Σχήμα 27

Σχήμα 28

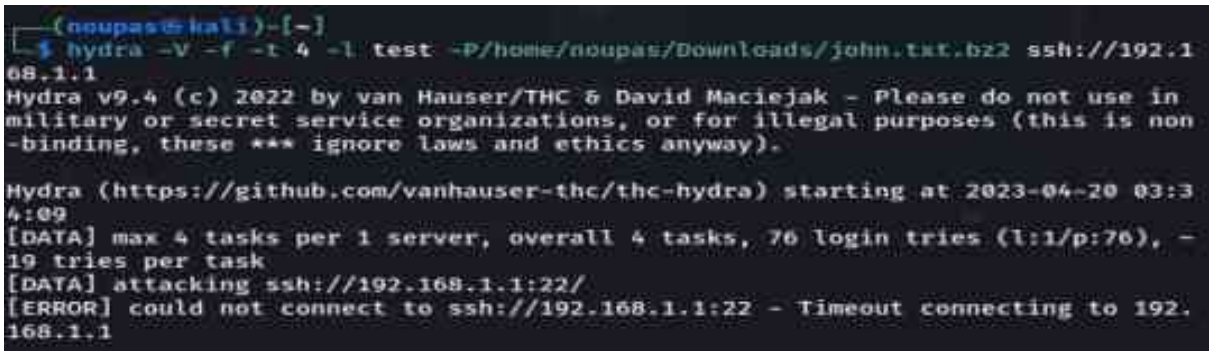
5.7.4.1 Brute Force Attack

Διότι χρησιμοποιήσαμε τυχαία παραγόμενα ονόματα και κωδικούς για τις μηχανές μας καθιστά απίστευτα χρονοβόρο για ένα brute force attack να μπορεί να σπάσει έγκαιρα αφού κάθε 30 μέρες οι κωδικοί αλλάζουν. Επιπλέον, χάρι στη χρήση ιδιωτικών και δημοσιών κλειδιών AES και RSA, αν ο επιτιθέμενος καταφέρει παρόλα αυτά να σπάσει τους κωδικούς των χρηστών, πάλι δεν μπορεί να ανταλλάξει μηνύματα με τον διακομιστή και επομένως δεν μπορεί να σπάσει την αλυσίδα αφού δεν έχει τα απαραίτητα κλειδιά. Επιπλέον, με την χρήση 256 bit κατακερματισμού (SHA-256) καθιστούμε αδύνατον με τα σημερινά δεδομένα να βρει ο επιτιθέμενος την τιμή κατακερματισμού που χρησιμοποιείται.

5.7.4.2 Dictionary Attack

Σε αυτό το πείραμα πήραμε μια από τις πιο γνωστές λίστες κωδικών την “john.txt.bz2” και μέσω του εργαλείου Hydra προσπαθήσαμε να πάρουμε πρόσβαση στο διακομιστή του πειράματος μέσω ssh. Το συγκεκριμένο εργαλείο δημιουργεί διεργασίες οι οποίες με στόχο την διεύθυνση του διακομιστή, δηλαδή ¨192.168.1.1¨ και την πύλη του πρωτοκόλλου ssh δηλαδή ¨22¨, δοκιμάζει όλους τους κωδικούς που περιέχει η λίστα προσπαθώντας να πάρει πρόσβαση. Όπως βλέπουμε και στο παρακάτω Σχήμα 29 η

προσπάθειά μας ήταν ανεπιτυχής καθώς επιλέχθηκε ένας δύσκολος κωδικός πρόσβασης. Επιπλέον, αξιοσημείωτο το γεγονός πως αν ο επιτιθέμενος καταφέρει να βρει τους κωδικούς πρόσβασης κάποιας μηχανής εξακολουθεί να μην έχει τα απαραίτητα κλειδιά έτσι ώστε να μπορεί να επικοινωνήσει με τον διακομιστή. Έτσι, το πρωτόκολλο καθίσταται ασφαλές ως προς επιθέσεις brute force.



Σχήμα 29

5.7.5 SQL Injection

Το πρωτόκολλο δεν κινδυνεύει από επιθέσεις SQL injection διότι πολύ απλά δεν χρησιμοποιεί SQL ή κάποια άλλη βάση δεδομένων για να φυλάσσει τα στοιχεία των χρηστών. Το πρωτόκολλο χρησιμοποιεί την τεχνολογία blockchain για να φυλάσσει τα δεδομένα των χρηστών. Ο διακομιστής δημιουργεί μία δική του αλυσίδα με μία φιλοσοφία εμπνευσμένη από τις πιο δημοφιλείς αλυσίδες blockchain, όπως το Ethereum. Μέχρι και σήμερα κανένας δεν μπόρεσε να σπάσει την τεχνολογία blockchain, έτσι μπορούμε να πούμε με βεβαιότητα ότι η βάση δεδομένων μας είναι ασφαλής από οποιαδήποτε επίθεση τύπου SQL injection. Στο παρακάτω Σχήματα 30 και 31 μπορούμε να δούμε τον τρόπο με τον οποίο ο διακομιστής ¨χτίζει¨ την αλυσίδα καθώς και τη στοιχεία χρησιμοποιεί για τον κατακερματισμό και στη συνέχεια για την ταυτότητα των μπλοκ στην αλυσίδα, καθώς και τον τρόπο που ελέγχεται η αλυσίδα σε κάθε νέα εγγραφή.

Σχήμα 30

Σχήμα 31

Κεφάλαιο 6
Συμπεράσματα

6.1 Επισκόπηση

Σε αυτό το παράρτημα θα διεξάγουμε ένα συμπέρασμα βάση των πειραμάτων που έγιναν και επιπλέον θα παρουσιάσουμε κάποιες προτάσεις ως προς ανάπτυξη του πρωτοκόλλου.

6.2 Συμπεράσματα

Μέσω της παρούσας διπλωματικής εργασίας μάθαμε πόσο σημαντική είναι η ασφαλής επικοινωνία μεταξύ των μηχανών και η ενεργειακή απόδοση στο διαδίκτυο των πραγμάτων. Επίσης, μέσω της ανάπτυξης του πρωτοκόλλου αυτού αναπτύξαμε και ενσύρματα και ασύρματα δίκτυα για της μηχανές μας. Μαζί με αυτό, είδαμε πως προγραμματίζονται και οι μηχανές Raspberry Pi, καθώς και το λειτουργικό σύστημα (Raspbian) που χρησιμοποιούν. Ακόμα, είδαμε τα διάφορα προβλήματα που μπορεί κανείς να αντιμετωπίσει στην ανάπτυξη ενός παρόμοιου πρωτοκόλλου, όπως η φθορά των καρτών SD που περιέχουν το λογισμικό των μηχανών ή το συγχρονισμό της ζώνης ώρας των μηχανών . Επιπλέον, είδαμε τις πιο γνωστές επιθέσεις που γίνονται σήμερα και πως μπορεί κανείς να προστατευτεί από αυτές. Σίγουρα στο μέλλον θα υπάρξουν πολλοί περισσότερες χρήσεις για μηχανές στο διαδίκτυο των πραγμάτων αφού ο τομέας αυτός ολοένα και αναπτύσσεται. Επιπρόσθετα, είδαμε πως η τεχνολογία blockchain μπορεί να μας προσφέρει μια ασφαλή βάση δεδομένων η οποία είναι και αποδοτική στην ανάρτηση δεδομένων. Τέλος, είδαμε όλα τα απαραίτητα εργαλεία έτσι ώστε να στηθεί ένα τέτοιο πρωτόκολλο καθώς και το δίκτυο επικοινωνίας τους.

6.3 Μελλοντική Δουλειά

Στη παρούσα διπλωματική εργασία στρέψαμε την προσοχή μας στην απόδοση και την ασφάλεια ενός πρωτοκόλλου που με την χρήση της τεχνολογίας blockchain προσφέρει ασφαλή επικοινωνία μεταξύ των μηχανών στο δίκτυο των πραγμάτων. Σκοπός της διπλωματικής εργασίας ήταν να προσφέρει ασφαλή επικοινωνία με την χρήση υφιστάμενων αλγορίθμων κρυπτογράφησης (RSA,AES) και κατακερματισμού (SHA-256). Επιπλέον ,διερευνήθηκε εις βάθος η τεχνολογία blockchain η οποίοι εφαρμόστηκε στη συνέχεια για να έχουμε ένα ασφαλής μέρος για να αποθηκεύουμε τα στοιχεία των πελατών/χρηστών του συστήματος. Το οποίο στη συνέχεια, μας επιτρέπει να έχουμε ένα αποδοτικό τρόπο επικοινωνίας μεταξύ των μηχανών. Αποδοτικό, τόσο ως προς εξοικονόμηση ενέργειας (η οποία είναι κρίσιμη για συστήματα IoT) , αλλά και γρήγορης ανάκτησης δεδομένων (ώστε να χρειαζόμαστε μικρή υπολογιστική δύναμη) ενώ ταυτόχρονα η επικοινωνία μεταξύ των μηχανών είναι εξακριβωμένα ασφαλής αλλά και γρήγορη.

Επιπρόσθετα, θα ήθελα και εγώ με τη σειρά μου, να προτείνω, στο μέλλον να γίνει ενσωμάτωση κάποιας υφιστάμενης αλυσίδας π.χ. Ethereum [13], ώστε να επωφεληθεί πλήρως το πρωτόκολλο επικοινωνίας τόσο από την απόδοση όσο και για επιπλέον λειτουργίες και δυνατότητες που προσφέρουν τέτοιες αλυσίδες.

Επιπλέον, θα ήθελα να προτείνω στο μέλλον να προστεθούν και άλλοι διακομιστές για περισσότερη ασφάλεια (αφού δεν θα υπάρχει πλέον ένα σημείο συμφόρησης) και για περισσότερη απόδοση σε περίπτωση που προστεθούν και άλλες μηχανές (πελάτες).

Τελευταίο, αλλά εξίσου σημαντικό είναι να αυτοματοποιηθεί η εγκατάσταση, η κλιμάκωση αλλά και η διαχείριση της εφαρμογής του πρωτοκόλλου. Επιπλέον, η εγκατάσταση των απαραίτητων βιβλιοθηκών αλλά και η ρύθμιση αρχείων που χρειάζεται το πρωτόκολλο όπως τη ρύθμιση του δικτύου, της ζώνης ώρας και άλλα που μπορεί να χρειαστεί στο μέλλον, μπορούν όλα αυτά να αυτοματοποιηθούν με την χρήση conditioner από ένα σύστημα ανοιχτού κώδικα όπως το Kubernetes[15].

Βιβλιογραφία

[1] Arab Computer Society, IEEE Computer Society, and Institute of Electrical and Electronics Engineers., *2018 IEEE/ACS 15th International Conference on Computer Systems and Applications : October 28th to November 1st, 2018, Aqaba, Jordan*.

[2] M. T. Hammi, B. Hammi, P. Bellot, A. Serhrouchni, T. Ltci, and F. Paristech, "Bubbles of Trust: a decentralized Blockchain-based authentication system for IoT." [Online]. Available: https://cointelegraph.com/news/proof-of-importance-nem-is-going-to-add-

[3] A. Z. Ourad, B. Belgacem, and K. Salah, "Using blockchain for IOT access control and authentication management," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Springer Verlag, 2018, pp. 150–164. doi: 10.1007/978-3-319-94370-1_11.

[4] A. Chaudhary, S. K. Peddoju, and K. Kadarla, "Study of Internet-of-Things Messaging Protocols Used for Exchanging Data with External Sources," in *Proceedings - 14th IEEE International Conference on Mobile Ad Hoc and Sensor Systems, MASS 2017*, Institute of Electrical and Electronics Engineers Inc., Nov. 2017, pp. 666–671. doi: 10.1109/MASS.2017.85.

[5] V. Sarafov Advisor and J. Seeger, "Comparison of IoT Data Protocol Overhead", doi: 10.2313/NET-2018-03-1_02.

[6] SKR Engineering College, Institute of Electrical and Electronics Engineers. Madras Section, and Institute of Electrical and Electronics Engineers, *International Conference on Energy, Communication, Data Analytics & Soft Computing (ICECDS) - 2017 : 1st & 2nd August 2017*.

[7] L. Năstase, I. Sandu, and N. Popescu, "An experimental evaluation of application layer protocols for the internet of things," *Studies in Informatics and Control*, vol. 26, no. 4, pp. 403–412, 2017, doi: 10.24846/v26i4y201704.

[8] Y. Chen and T. Kunz, "Performance evaluation of IoT protocols under a constrained wireless access network," in *2016 International Conference on Selected Topics in Mobile and Wireless Networking, MoWNeT 2016*, Institute of Electrical and Electronics Engineers Inc., Jun. 2016. doi: 10.1109/MoWNet.2016.7496622.

[9] I. Hedi, I. Špeh, and A. Šarabok, "IoT network protocols comparison for the purpose of IoT constrained networks," in *2017 40th International Convention on Information and Communication Technology, Electronics and Microelectronics, MIPRO 2017 - Proceedings*, Institute of Electrical and Electronics Engineers Inc., Jul. 2017, pp. 501–505. doi: 10.23919/MIPRO.2017.7973477.

[10] N. ACIT 4. 2016 Las Vegas *et al.*, *ACIT-CSII-BCD 2016 4th International Conference on Applied Computing and Information Technology (ACIT 2016), 3rd International Conference on Computational Science/Intelligence and Applied Informatics (CSII 2016), 1st International Conference on Big Data, Cloud Computing, Data Science & Engineering (BCD 2016) : 12-14 December 2016, Las Vegas, Nevada, USA : proceedings*.

[11] E. Al-Masri *et al.*, “Investigating Messaging Protocols for the Internet of Things (IoT),” *IEEE Access*, vol. 8, pp. 94880–94911, 2020, doi: 10.1109/ACCESS.2020.2993363.

[12] S. Nakamoto, “Bitcoin: A Peer-to-Peer Electronic Cash System.” [Online]. Available: www.bitcoin.org

[13] V. Buterin, “Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform.”

[14] B. Chase and E. MacBrough, “Analysis of the XRP Ledger Consensus Protocol,” Feb. 2018, [Online]. Available: http://arxiv.org/abs/1802.07242

[15] “GitHub - kubernetes_kubernetes_ Production-Grade Container Scheduling and Management”.