ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ
ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Drone digital video storage and secure transmission for Raspberry Pi

Ψηφιακή αποθήκευση και ασφαλής μετάδοση βίντεο με τη χρήση Raspberry Pi για drone

Ευάγγελος Κίτσος
Α.Μ.: 711161121

Εισηγητής:
Βογιατζής Ιωάννης (επιβλέπων)

Αθήνα, Μάρτιος 2024

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Drone digital video storage and secure transmission for Raspberry Pi

Ψηφιακή αποθήκευση και ασφαλής μετάδοση βίντεο με τη χρήση Raspberry Pi για drone

Ευάγγελος Κίτσος
Α.Μ.: 711161121

Εισηγητής:
Βογιατζής Ιωάννης

Ioannis Voyiatzis
Digitally signed by Ioannis Voyiatzis
Date: 2024.04.16 14:35:26 +03'00'

Εξεταστική επιτροπή:
[1] Stavros Fatouros
Digitally signed by Stavros Fatouros
Date: 2024.04.16 22:56:59 +03'00'

[2] GEORGIOS LENTARIS
Digitally signed by GEORGIOS LENTARIS
Date: 2024.04.17 10:09:30 +03'00'

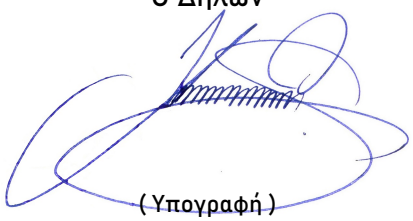
Ημερομηνία εξέτασης:
[...]

ΔΗΛΩΣΗ ΣΥΓΓΡΑΦΕΑ ΔΙΠΛΩΜΑΤΙΚΗΣ ΕΡΓΑΣΙΑΣ

Ο κάτωθι υπογεγραμμένος Ευάγγελος Κίτσος του Χρήστου, με αριθμό μητρώου 711161121 φοιτητής του Πανεπιστημίου Δυτικής Αττικής της Σχολής Μηχανικών του Τμήματος Μηχανικών Πληροφορικής και Υπολογιστών, δηλώνω υπεύθυνα ότι:

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

Ο Δηλών



(Υπογραφή)

ΕΥΑΓΓΕΛΟΣ ΚΙΤΣΟΣ
Προπτυχιακός Φοιτητής
Τμήματος Μηχανικών Πληροφορικής και Υπολογιστών

ΕΥΧΑΡΙΣΤΙΕΣ

Με την ολοκλήρωση της παρούσας διπλωματικής εργασίας ολοκληρώνεται ο προπτυχιακός κύκλος σπουδών στη Σχολή Μηχανικών Πληροφορικής και Υπολογιστών του Πανεπιστημίου Δυτικής Αττικής. Με την ευκαιρία αυτή θα ήθελα να ευχαριστήσω ορισμένα άτομα που με στήριξαν και χρήζουν ιδιαίτερης μνείας.

Αρχικά, τον κ. Ψιλιά Δημήτριο, υποψήφιο διδάκτορα στη Σχολή Μηχανικών Πληροφορικής και Υπολογιστών, για την πολλή υπομονή που μου επέδειξε κατά τη συγγραφή της διπλωματικής μου εργασίας και τις άμεσες και πολύτιμες συμβουλές του που με ώθησαν με ουσιαστικό τρόπο να διασαφηνίσω πολλά πράγματα εύκολα και άμεσα, όταν το είχα ανάγκη.

Θα ήθελα να ευχαριστήσω τον καθηγητή κ. Μελετίου Γεώργιο για τη μεταδοτικότητά του και τις γνώσεις πάνω στο αντικείμενο της Επιστήμης της Πληροφορικής που μου προσέφερε κατά τη διάρκεια φοίτησής μου.

Ένα μεγάλο ευχαριστώ θα ήθελα να απευθύνω στη σύζυγό μου, Κατερίνα, που με την υπομονή της, την απασχόλησή της με οικογενειακά θέματα καθημερινότητας και την μεγάλη ψυχολογική ώθηση και την υποστήριξή της, έγινε δυνατή η ενασχόλησή μου με το αντικείμενο της Πληροφορικής και την επιστημονική εργασία.

Επίσης ένα μεγάλο ευχαριστώ αξίζει η αδερφή μου, Ειρήνη, για την ψυχολογική υποστήριξη που μου προσέφερε, τη βοήθεια σε πολλές τεχνικές λεπτομέρειες και τη διάθεση και την ενέργεια που μου κινητοποίησε μέσα από τις συζητήσεις μας.

Εξέχουσα θέση κατέχουν οι γονείς μου στους οποίο θα ήθελα να απευθύνω το μεγαλύτερο ευχαριστώ γι' αυτό που είναι, την πηγή έμπνευσης που απορρέουν από όσα μου έχουν πει αδιάκοπα και το άθραυστο πρότυπο που αποτελούν για εμένα μέχρι σήμερα.

ΠΕΡΙΛΗΨΗ

Από την αρχή της ανάπτυξης του πολιτισμού και της γραφής μέχρι και σήμερα δημιουργήθηκε η ανάγκη του περιορισμού της ροής πληροφοριών μεταξύ ομάδων ατόμων. Από την απλή απόκρυψη μηνυμάτων μέχρι την ολοκληρωτική τους κρυπτογράφηση, ο άνθρωπος εκτός από τη σωστή διανομή των πληροφοριών επιθυμούσε παράλληλα να εξαιρέσει συγκεκριμένες ομάδες από τη γνώση τους. Ακόμη περισσότερο, με την εκρηκτική ανάπτυξη και δημοφιλή αποδοχή των κινητών συσκευών και του Διαδικτύου, ο τρομερός όγκος πληροφορίας που μετακινούνταν ανά πάσα στιγμή αναζητούσε τρόπους σωστής διαχείρισης.

Η κρυπτογράφηση δεδομένων δεν είναι κάτι καινούργιο. Από την αρχαιότητα μαίνεται η αντιπαράθεση μεταξύ κρυπτογράφων και κρυπταναλυτών, με τους μεν να προσπαθούν να διασφαλίσουν τα μηνύματα που θα ήθελαν να διανείμουν και με τους δε να εντείνουν τις προσπάθειές τους στην ανάλυση και τελική αποκάλυψή τους. Η επιστήμη της Κρυπτογραφίας εξελίχθηκε σε ιδιαίτερα δύσκολη και απαιτεί στιβαρό μαθηματικό υπόβαθρο και από τις δύο πλευρές.

Στην πορεία αυτή, πολλά συστήματα και πολλοί αλγόριθμοι έχουν χρησιμοποιηθεί (και παρωχηθεί) για να κρατήσουν τα μηνύματα ασφαλή. Ο αλγοριθμος Rijndael (των Rijmen και Daemen), γνωστός ευρύτερα και ως AES (Advanced Encryption System) χρησιμοποιείται σήμερα ευρέως. Απαντάται σε κλειδιά router, σε βίντεο και εικόνες, σε ηχητικά μηνύματα και συνομιλίες αλλά και σε βάσεις δεδομένων.

Η παρούσα διπλωματική εργασία επικεντρώνει το ενδιαφέρον της στη μελέτη της ασφαλή μετάδοσης δεδομένων ανάμεσα σε δύο υπολογιστές. Ειδικότερα, θα χρησιμοποιηθούν Single Board Computers Raspberry Pi, ένας εκ των οποίων προορίζεται να μεταφέρεται από drone και θα είναι εξοπλισμένος με κάμερα. Η κάμερα θα καταγράφει βίντεο το οποίο με τη δημιουργία του θα κρυπτογραφείται, με τα δύο αρχεία να παραμένουν στη συσκευή. Το κρυπτογραφημένο αρχείο θα μεταδίδεται μέσω socket στο σταθμό εδάφους, ο οποίος θα είναι ο άλλος υπολογιστής Raspberry Pi, που θα αποκρυπτογραφεί και θα αναπαράγει το βίντεο που αρχικώς κινηματογραφήθηκε.

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ:
Raspberry Pi, κρυπτογράφηση, ασφαλής μετάδοση, Python, Μη Επανδρωμένα Αεροσκάφη.

ABSTRACT

Since the beginning of the development of civilization and scripture until today, the need for restriction of the flow of information between groups of people emerged. From the simple hiding of messages up to their complete encryption, human desired, apart from the correct delivery of information, the exclusion of certain groups from their knowledge. Furthermore, with the explosive development and wide reception of mobile devices and the Internet, the massive amount of data trafficked at any given moment needed to have ways of correct management.

Data encryption is not something new. From ancient times a race is raging between cryptographers and decipherers, the first trying to secure the messages they wanted to distribute and the latter to strengthen their efforts into the cryptanalysis and their eventual revelation. The science of Cryptography is extremely difficult and demands a robust mathematical background from both sides.

During these times, many systems and many cryptociphers have been used (and also become obsolete) to keep messages safe. The Rijndael algorithm (written by Rijmen and Daemen), alternatively known as AES (Advanced Encryption Standard) is used widely today. It has been used in router keys, video and images, sound files, conversations and databases.

This diploma thesis focuses on the study of secure transmission of data between two computers. In particular, two single board Raspberry Pi computers are going to be used, one of which is designed to be mounted onto a drone and equipped with a camera. That camera will record video that will be encrypted after its creation, with both files remaining on the device. The encrypted file will be transmitted through sockets to the ground station which will be another Raspberry Pi computer that will decrypt and play the video which was previously encrypted.

KEY WORDS:
Raspberry Pi, encryption, secure transmission, Python, unmanned aerial vehicles.

ΠΕΡΙΕΧΟΜΕΝΑ

ΕΥΡΕΤΗΡΙΟ ΕΙΚΟΝΩΝ

ΚΕΦΑΛΑΙΟ 1
ΕΙΣΑΓΩΓΗ

1.1 - ΑΝΤΙΚΕΙΜΕΝΟ ΤΗΣ ΔΙΠΛΩΜΑΤΙΚΗΣ ΕΡΓΑΣΙΑΣ

Με την τεχνολογική έκρηξη των τελευταίων δεκαετιών έγιναν εμπορικά διαθέσιμα στο ευρύ κοινό υπολογιστικά συστήματα και συσκευές που μέχρι πρόσφατα απευθύνονταν σε εταιρείες ή ερευνητικά εργαστήρια. Άμεσο αποτέλεσμα ήταν γεωμετρική αύξηση του ρυθμού μετάδοσης πληροφοριών μεταξύ τους, που άτυπα μεν αλλά επιτακτικά δε, απαίτησε την ασφαλή μετάδοσή τους.

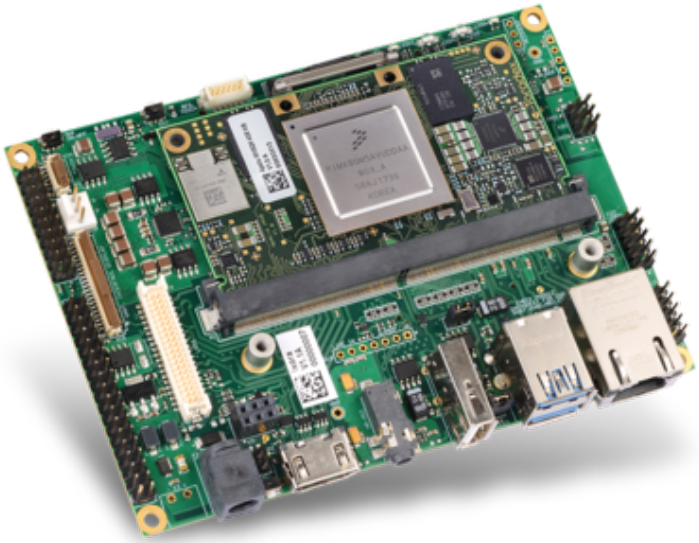
Με την ασφαλή μετάδοση νοείται η μετατροπή των δεδομένων πριν τη μεταφορά τους από το ένα σημείο στο άλλο ώστε να είναι ορατές και αξιοποιήσιμες μόνο από αυτούς για τους οποίους προορίζονται. Μια μέθοδος για να επιτευχθεί αυτή η μετατροπή και ο περιορισμός πρόσβασης είναι η κρυπτογράφηση. Η σημερινή χρήση της είναι εξαιρετικά ευρεία ‘ από εφαρμογές ανταλλαγής μηνυμάτων μέχρι αποθηκευμένα δεδομένα σε servers και δίσκους, ακόμη και σε επιθέσεις ransomware. Ο χρόνος για την κάθε φορά κρυπτογράφηση και αποκρυπτογράφηση θυσιάζεται αλλά εγγυάται την ασφάλεια κατά τη μετάδοση δεδομένων.

Η παρούσα Διπλωματική Εργασία έχει ως κύριο αντικείμενο την κρυπτογράφηση. Θα μεταδοθούν κρυπτογραφημένα αρχεία από μια συσκευή σε μια άλλη που θα τα αποκρυπτογραφήσει. Γι’ αυτό το σκοπό θα γίνει χρήση των Single Board Computers (Υπολογιστές Μονής Πλακέτας) που ονομάζονται Raspberry Pi. Ένα από αυτά θα συνδεθεί σε ένα router (που θα χρησιμοποιηθεί ως hotspot) και θα λειτουργήσει ως server αναμένοντας δεδομένα. Ένα δεύτερο Raspberry Pi θα συνδεθεί στο ίδιο router. Θα κινηματογραφήσει βίντεο το οποίο αμέσως θα κρυπτογραφήσει και ύστερα, ως client στον προαναφερθέντα server, θα του αποστείλει τα κρυπτογραφημένα δεδομένα. Ο server μόλις τα λάβει θα τα αποκρυπτογραφήσει. Έτσι, θα δειχθεί ότι ακόμα κι αν ένα μη εξουσιοδοτημένο άτομο αποκτήσει πρόσβαση στα μεταδιδόμενα δεδομένα, αυτά θα είναι μη αξιοποιήσιμα.

Για την επίτευξη του σκοπού αυτού αναπτύχθηκε κώδικας σε Python και αυτοματοποίηση μέσω του κελύφους Bash. Η εφαρμογή σε πραγματικές συνθήκες της όλης υλοποίησης αφορά τη χρήση σε Μη Επανδεωμένα Αεροσκάφη (drones) αλλά απευθύνεται γενικά σε οποιαδήποτε εφαρμογή απαιτεί την ασφαλή μετάδοση δεδομένων.

1.2 - SINGLE BOARD COMPUTER

Το Raspberry Pi συγκαταλέγεται στους Single Board Computers (SBC). Πρόκειται για έναν ολοκληρωμένο υπολογιστή που τρέχει ένα πλήρες λειτουργικό σύστημα, με επεξεργαστή, κύρια μνήμη και έναν αριθμό από διεπαφές. Τα βασικά υποσυστήματά του είναι μη αφαιρούμενα και μη αναβαθμίσιμα, καθώς είναι ενσωματωμένα πάνω σε μία και μοναδική πλακέτα, συνήθως μικρού μεγέθους. Κατά κανόνα, οι υπολογιστές αυτοί διαθέτουν επίσης GPIO Pins (General Purpose Input/Output) που είναι μια σειρά από ακίδες γενικού σκοπού με σταθερές λειτουργίες η κάθε μία, ενώ η πρόσβαση σε αυτές γίνεται με κώδικα. Εκτός από το Raspberry Pi κυκλοφορούν στην αγορά πολλοί Single Board Computers, όπως το Asus Tinker Board, το Oderoid-N2+, το Nvidia Jetson Nano, και άλλα.



Εικόνα 1.1
Παράδειγμα Single Board Computer
toradex.com/computer-omn-modules/apalis-arm-family/nxp-freescale-imx-6

1.3 - PYTHON

Η Python είναι μια αντικειμενοστρεφής γλώσσα προγραμματισμού γενικού σκοπού, υψηλού επιπέδου. Είναι πολύ κοντά στην ανθρώπινη, με τους δημιουργούς και διαχειριστές της να στοχεύουν στην απλότητα, την αναγνωσιμότητα και την εύκολη αποσφαλμάτωση. Χαρακτηρίζεται από ταχύτητα μιας και δε χρησιμοποιεί μεταγλωττιστή αλλά διερμηνευτή. Η δημοτικότητά της είχε ως αποτέλεσμα την προσέλκυση προγραμματιστών και εργαστηρίων, και τελικώς την ενσωμάτωση αναρίθμητων πακέτων που αυτοματοποιούν διαδικασίες. Ακόμη, η Python μπορεί να ενσωματώσει κώδικα γραμμένος σε άλλες γλώσσες προγραμματισμού καθώς και να ενσωματωθεί η ίδια σε εφαρμογές γραμμένες σε άλλες γλώσσες.

1.4 - ΔΙΑΡΘΡΩΣΗ ΔΙΠΛΩΜΑΤΙΚΗΣ

Η διάρθρωση της παρούσας Διπλωματικής εργασίας έχει ως εξής:

Στο δεύτερο κεφάλαιο γίνεται μια παρουσίαση των drones, των κατηγοριών τους, των μερών που τα αποτελούν, καθώς και τους περιορισμούς που υφίστανται στην πτήση και κινηματογράφηση με drone.

Στο τρίτο κεφάλαιο παρουσιάζεται το βασικό υλικό της εργασίας, το Raspberry Pi. Αναφέρονται οι εκδόσεις του, τα χαρακτηριστικά του μοντέλου που χρησιμοποιήθηκε, διάφορες λειτουργίες ενσωματωμένες σε αυτό, η αρχιτεκτονική του επεξεργαστή του καθώς και διάφορα εναλλακτικά συστήματα μονής πλακέτας.

Στο τέταρτο κεφάλαιο καλύπτεται η κρυπτογραφία ως έννοια, τα κριτήρια ενός ορθού αλγόριθμου κρυπτογράφησης και οι αρχές που αξιολογούν την ισχύ του. Επίσης γίνεται επεξήγηση του αλγορίθμου της εργασίας (AES), των τρόπων λειτουργίας του και του συγκεκριμένου που χρησιμοποιήθηκε (CFB).

Στο πέμπτο κεφάλαιο παρουσιάζεται το λειτουργικό σύστημα πάνω στο οποίο βασίζεται η διανομή που τρέχει το Raspberry Pi, και τα κελύφη επικοινωνίας του χρήστη με τον πυρήνα του.

Στο έκτο κεφάλαιο παρουσιάζεται η γλώσσα προγραμματισμού Python, που χρησιμοποιήθηκε για την ανάπτυξη του βασικού μέρους του κώδικα. Επίσης, αναφέρονται δύο

πακέτα με βιβλιοθήκες που έχουν ενσωματωθεί στην Python. Το OpenCV (Computer Vision) που αφορά Όραση Υπολογιστή και χειρίζεται εικόνα και βίντεο. Και τα Sockets, που δημιουργούν μοναδικές συνδέσεις μεταξύ υπολογιστών για τη λήψη και αποστολή δεδομένων μεταξύ τους.

Στο έβδομο κεφάλαιο παρουσιάζεται το hardware που απαιτήθηκε συγκεκριμένα για την υλοποίηση του ζητούμενου της εργασίας. Μετρήθηκε επιπρόσθετα και το βάρος του client για την κατάλληλη επιλογή drone σε περίπτωση ενσωμάτωσης σε αυτό.

Στο όγδοο κεφάλαιο αναλύεται διεξοδικά ο κώδικας που τρέχει ο client και ο server, καθώς και η λειτουργία των επιμέρους τμημάτων λογισμικού που αναπτύχθηκαν.

Στο ένατο κεφάλαιο πραγματοποιείται δοκιμή βήμα προς βήμα όλων των διαδικασιών για τη σύνδεση των client και server στο router που λειτουργεί ως hotspot και η ασφαλής μετάδοση βίντεο από το ένα στο άλλο.

Και τέλος, στο δέκατο κεφάλαιο παρουσιάζονται τα συμπεράσματα που απορρέουν από την υλοποίηση αυτή καθώς και προεκτάσεις του παρόντος συστήματος σε άλλους τομείς.

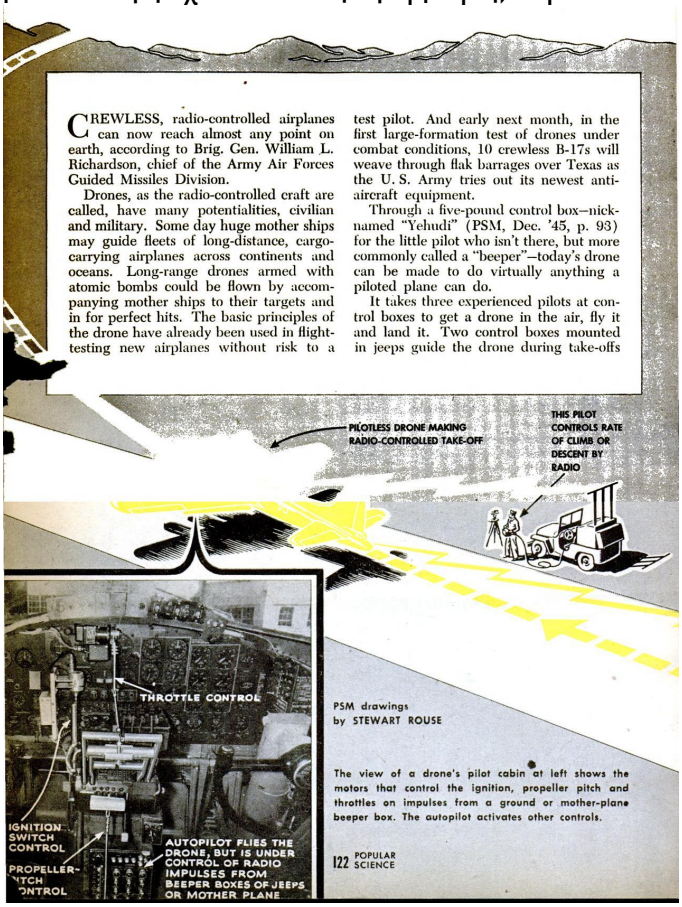
ΚΕΦΑΛΑΙΟ 2
ΜΗ ΕΠΑΝΔΡΩΜΕΝΑ ΑΕΡΟΣΚΑΦΗ

2.1 - ΕΙΣΑΓΩΓΗ

Τα Μη Επανδρωμένα Αεροσκάφη (ΜΕΑ), ονομαζόμενα επίσημα από την Ελληνική Πολεμική Αεροπορία ως Συστήματα Μη Επανδρωμένων Αεροσκαφών (ΣμηΕΑ), χρησιμοποιούνται σε αναρίθμητα σενάρια και κατηγοριοποιούνται σε πολλούς τύπους ανάλογα με τα χαρακτηριστικά τους. Στην καθομιλουμένη αναφέρονται ως *drones* ενώ η επίσημη μετάφραση από τα ελληνικά στα αγγλικά είναι *δρόνοι*, σύμφωνα με την ΕΛ.ΕΤ.Ο. (Ελληνική Εταιρεία Ορολογίας, 2022). Το Κέντρο Ερεύνης Επιστημονικών Όρων και Νεολογισμών της Ακαδημίας Αθηνών είχε προτείνει τον όρο *τηλεσκάφος,* που όμως έχει παραγκωνιστεί.

Η αγγλική ονομασία, που σημαίνει κηφήνας, δάνεισε την έννοιά της στο μη επανδρωμένο αεροσκάφος οποιουδήποτε είδους λόγω του ήχου των πρώτων παλιών στρατιωτικών υλοποιήσεων του είδους που έμοιαζε με βόμβο και στο ότι φαίνονταν να πετούν μόνα τους. Ο όρος drone δεν έγινε εύκολα αποδεκτός. Μάλιστα, συνάντησε σθεναρή αντίσταση από τους επαγγελματίες της αεροπορίας, από τις επιτροπές ορολογίας αλλά και από τις κυβερνητικές αρχές (Σαραντάκος, 2017).

Τα drones δε χρησιμοποιούνται μόνο για στρατιωτικούς σκοπούς αν και η χρήση τους ξεκίνησε γι' αυτό το λόγο. Στο τεύχος Νοεμβρίου 1946 του περιοδικού Popular Science υπάρχει γραπτή αναφορά για "χωρίς πλήρωμα ραδιοελεγχόμενα αεροπλάνα που μπορούν να φτάσουν σε οποιοδήποτε σημείο του πλανήτη" και "τα drones, όπως ονομάζονται αυτά τα ελεγχόμενα με ραδιοκύματα σκάφη έχουν πολλές εφαρμογές, στρατιωτικές αλλά και πολιτικές".


CREWLESS, radio-controlled airplanes can now reach almost any point on earth, according to Brig. Gen. William L. Richardson, chief of the Army Air Forces Guided Missiles Division.

Drones, as the radio-controlled craft are called, have many potentialities, civilian and military. Some day huge mother ships may guide fleets of long-distance, cargo-carrying airplanes across continents and oceans. Long-range drones armed with atomic bombs could be flown by accompanying mother ships to their targets and in for perfect hits. The basic principles of the drone have already been used in flight-testing new airplanes without risk to a test pilot. And early next month, in the first large-formation test of drones under combat conditions, 10 crewless B-17s will weave through flak barrages over Texas as the U. S. Army tries out its newest anti-aircraft equipment.

Through a five-pound control box—nicknamed "Yehudi" (PSM, Dec. '45, p. 93) for the little pilot who isn't there, but more commonly called a "beeper"—today's drone can be made to do virtually anything a piloted plane can do.

It takes three experienced pilots at control boxes to get a drone in the air, fly it and land it. Two control boxes mounted in jeeps guide the drone during take-offs

PILOTLESS DRONE MAKING RADIO-CONTROLLED TAKE-OFF

THIS PILOT CONTROLS RATE OF CLIMB OR DESCENT BY RADIO

THROTTLE CONTROL

IGNITION SWITCH CONTROL

PROPELLER-PITCH CONTROL

AUTOPILOT FLIES THE DRONE, BUT IS UNDER CONTROL OF RADIO IMPULSES FROM BEEPER BOXES OF JEEPS OR MOTHER PLANE

PSM drawings by STEWART ROUSE

The view of a drone's pilot cabin at left shows the motors that control the ignition, propeller pitch and throttles on impulses from a ground or mother-plane beeper box. The autopilot activates other controls.

122 POPULAR SCIENCE

Εικόνα 2.1

Απόκομμα από το τεύχος Νοεμβρίου 1946 του περιοδικού Popular Science
https://books.google.gr/books?id=_CADAAAAMBAJ&pg=PA122&dq=%22drone%22&hl=en&sa=X&ei=8rcpVZiONcb1oATH34GoCg&redir_esc=y#v=onepage&q&f=false

Μια συνήθης εφαρμογή των drones στις σημερινές δραστηριότητες είναι η διανομή αλληλογραφίας ή προμηθειών ανάγκης ταχύτατα, ακόμα και σε απρόσιτες περιοχές. Άλλη δημοφιλής χρήση τους είναι στην καταγραφή βίντεο, αφενός μεν σε επαγγελματικό επίπεδο για τις ανάγκες κινηματογραφικής ταινίας, αφετέρου δε σε ερασιτεχνικό επίπεδο με αυτόματη ακολούθηση χειριστή (αυτοκινηματογράφηση). Στον τομέα της ψυχαγωγίας, διοργανώνοται αγώνες επιδείξεων και ταχύτητας, ενώ συνεργαζόμενες μονάδες δημιουργούν συστάδες που σχηματίζουν παραστάσεις στον ουρανό. Ακόμη, δεν αποκλείονται και οι παράνομες ενέργειες όπως η μεταφορά ναρκωτικών ή άλλων παράνομων υλικών. Οι επιδέξιοι χειριστές drones που το οδηγούν με μεγάλη ταχύτητα σε στενή διαδρομή που απαιτεί λεπτομέρεια είναι ιδιαίτερα περιζήτητοι.

2.2 - ΚΑΤΗΓΟΡΙΟΠΟΙΗΣΗ

Ένας τυπικός διαχωρισμός είναι σε drones με φτερά σταθερά (fixed), περιστρεφόμενα (rotary) και που φτερουγίζουν (flapping). Με βάση το βάρος, υπάρχουν μικρά που ζυγίζουν κάτω από 1 κιλό (micro), μινιατούρες με βάρος από 1 έως 25 κιλά (miniature) και βαρέα πανω από 25 κιλά (heavy). Με βάση το χρόνο πτήσης, κατηγοριοποιούνται σε μικρού (κάτω από 5 ώρες ή 100 χιλιόμετρα), μεσαίου (από 5 έως 24 ώρες ή από 100 έως 400 χιλιόμετρα) και μεγάλους βελησεκούς (πάνω από 24 ώρες ή 1500 χιλιόμετρα). Μια ακόμη κατηγοριοποίση γίνεται και με το ύψος πτήσης. Τα drones χαμηλής πτήσης κυμαίνονται στα χαμηλού ύψους (μέχρι 1 χιλιόμετρο), μεσαίου ύψους (από 1 μέχρι 10 χιλιόμετρα) και μεγάλου ύψους (ανω των 10 χιλιομέτρων). Σημειώνεται ότι εκτός από την κίνηση στον αέρα, υπάρχουν drone που κινούνται πάνω ή και μέσα στο νερό. Έτσι, υπάρχουν τα επιφανείας (seaplane), υποβύχιας εκκίνησης (submarine launched) και υποβρύχια (underwater) (Xingbang, 2022). Στην παρούσα Διπλωματική εργασία νοούνται ως drones αυτά που αιωρούνται με έλικες.

RTF (Ready To Fly) – Έτοιμο για πτήση

Αυτά είναι τα πιο απλά, ο ορισμός των οποίων είναι αυτοεπεξηγηματικός. Σημαίνει ότι μπορούν να πετάξουν σχεδόν άμεσα, με την αφαίρεση της συσκευασίας. Προτείνονται ιδιαίτερα για αρχάριους διότι δεν απαιτούνται σχεδόν καθόλου αρχικές ρυθμίσεις πέρα από τη σύνδεση με το χειριστήριο και τη φόρτιση των μπαταριών.

Εικόνα 2.2
Intel 954400 Ready to Fly Drone
https://www.okdo.com/p/microbit-drone/

BNF (Bind N Fly) – Σύνδεση και πτήση

Ο όρος *σύνδεση* αναφέρεται στο ότι πωλούνται ως ολοκληρωμένα UAV αλλά χωρίς χειριστήριο. Είναι ίσως ένας τρόπος την εξοικονόμηση του κόστους του και την επαναχρησιμοποίηση κάποιου παλιού χειριστηρίου. Η συμβατότητα δεν είναι εγγυημένη όμως θεωρητικά το ίδιο χειριστήριο μπορεί να συνδεθεί με την πλειονότητα των εμπορικών drone (στην ίδια συχνότητα). Αυτό το είδος απευθύνεται σε έναν πιο πεπειραμένο χρήστη.

Εικόνα 2.3
Emax Tinyhawk Indoor FPV Racing Drone
https://www.myfpvstore.com/whoop-drones/ready-to-fly/emax-tinyhawks-75mm-1s-2s-micro-indoor-fpv-racing-drone-bnf/

ARF (Almost Ready To Fly) – Σχεδόν έτοιμο για πτήση

Εδώ χρειάζεται προεργασία πριν την πτήση. Η συσκευασία περιλαμβάνει το βασικό κορμό, ορισμένες φορές χωρίς τη μηχανή, και σίγουρα χωρίς το χειριστήριο / ελεγκτή πτήσης. Δεν απευθύνονται σε αρχάριους αλλά σε όσους επιθυμούν να ασχοληθούν με την κατασκευή και πτήση εξατομικευμένου drone, ακόμη και χρησιμοποιώντας μέρη από άλλο μηχάνημα.

Εικόνα 2.4
Mini Fly QuadCopter Drone ARF with MWC Board Brushless Motor

https://www.xheli.com/05h100-quadcopter-yellow-arf.html

2.3 – MEPH DRONE

Κέλυφος

Το περίβλημα χρησιμεύει στην προστασία του εσωτερικού του από τις πτώσεις. Ακόμη, τα φανταχτερά χρώματα βοηθούν στον οπτικό εντοπισμό του από απόσταση.

Έλικες

Οι έλικες έχουν ελαφρώς καμφθεί ώστε να έχουν κλίση από το επίπεδο και παρόμοιο με φτερά αεροπλάνου. Το σχήμα τους είναι που, περιστρεφόμενο, δίνει την ώθηση ανόδου.

Μπαταρία

Η μπαταρία είναι το μοναδικό εξάρτημα με περιορισμό. Μεγάλη μπαταρία σημαίνει αυξημένη ώρα πτήσης αλλά και μεγάλο βάρος, που την περιορίζει. Μια τυπική διάρκεια πτήσης μέχρι την εξάντλησή της στα drones του εμπορίου είναι τα 20 με 30 λεπτά.

Μηχανισμός προσγείωσης

Σε απλά drones είναι μαλακά λάστιχα στο κάτω μέρος των βραχιόνων. Άλλα έχουν ξεχωριστό αποσπώμενο μέρος που χρησιμεύει ώστε στην προσγείωση να κάθεται στο έδαφος χωρίς να βλαφθούν οι έλικες ή το σώμα.

Σώμα

Το κεντρικό σημείο του drone είναι αυτός ο ενισχυμένος άξονας. Όλα τα φυσικά στοιχεία του συνδέονται με αυτό, στο οποίο είναι και το κέντρο βάρους του.

Βραχίονες

Οι βραχίονες εκτείνονται από το κέντρο προς τα έξω και καταλήγουν στους έλικες. Πρέπει αφενός μεν να είναι ανθεκτικοί ώστε να υποστηρίζουν το όλο κατασκεύασμα και αφετέρου δε να είναι λεπτοί ώστε να μην εμποδίζουν την ώθηση από τους έλικες. Το μήκος τους επίσης συμβάλλει στην σταθερότητα και την ευελιξία της κίνησης.

Ελεγκτής πτήσης

Ο χρήστης έχει τον έλεγχο κίνησης, αλλά ο ελεγκτής εκτελεί μικροδιορθώσεις ώστε το drone να τηρεί μια σταθερή πτήση κόντρα στον άνεμο και τις πιέσεις.

GPS

Ορισμένα συστήματα είναι εξοπλισμένα με ελεγκτή GPS, που τους επιτρέπει να γνωρίζουν και να ορίζουν την τοποθεσία τους μέσω εμπορικών δορυφόρων που βρίσκονται σε τροχιά γύρω από τη Γη. Απαιτούνται 3 δορυφόροι για την επιτυχή διακρίβωση της τοποθεσίας του. Οι εμπορικοί δορυφόροι παρουσιάζουν ένα σφάλμα περίπου 5 μέτρων, ενώ οι στρατιωτικοί δορυφόροι έχουν ακρίβεια με μηδενικό σφάλμα απόστασης. Με το GPS είναι δυνατές λειτουργίες όπως η ανάγνωση θέσης, η αυτόνομη πτήση, η ακολούθηση προγραμματισμένης διαδρομής και η επιστροφή σε προκαθορισμένο σημείο.

Κάμερα

Η κάμερα μεταδίδει στο χειριστή ζωντανή εικόνα σε πραγματικό χρόνο. Η εικόνα μπορεί να μεταδίδεται σε ειδικό χειριστήριο ή σε smartphone (η οποία συνήθως απαιτεί εγκατάσταση εφαρμογής του κατασκευαστή). Μέσα από διάφορες λειτουργίες λογισμικού είναι δυνατή η τοποθέτηση φίλτρων, η αναγνώριση και ακολούθηση του χειριστή, ακόμη και η τρισδιάστατη θέαση βίντεο σε πρώτο πρόσωπο.

2.4 – ΝΟΜΙΚΑ ΖΗΤΗΜΑΤΑ ΔΕΔΟΜΕΝΩΝ ΚΙΝΗΜΑΤΟΓΡΑΦΗΣΗΣ ΚΑΙ ΠΤΗΣΗΣ

Σε αυτό το τμήμα παρατίθεται η κείμενη νομοθεσία που διέπει την κινηματογράφηση από drone και γενικά τα προσωπικά δεδομένα, σε εθνικό και ευρωπαϊκό επίπεδο, καθώς και τους περιορισμούς που διέπουν την πτήση των drones.

Στο άρθρο 370^{A} παρ. 2 του Ποινικού Κώδικα αναφέρεται: "Όποιος αθέμιτα παρακολουθεί με ειδικά τεχνικά μέσα ή αποτυπώνει σε υλικό φορέα προφορική συνομιλία μεταξύ τρίτων που δεν διεξάγεται δημόσια ή αποτυπώνει σε υλικό φορέα μη δημόσια πράξη άλλου, τιμωρείται με φυλάκιση τουλάχιστον ενός έτους. Με την ίδια ποινή τιμωρείται η πράξη του προηγούμενου εδαφίου και όταν ο δράστης αποτυπώσει σε υλικό φορέα το περιεχόμενο της συνομιλίας του με άλλον χωρίς τη ρητή συναίνεση του τελευταίου." Αυτό σημαίνει ότι απαγορεύεται όχι μόνο η αποθήκευση βίντεο (και γενικώς υλικού) αλλά και η παρακολούθηση γεγονότων που διαδραματίζονται σε ιδιωτικό επίπεδο χωρίς τη συναίνεση όλων όσων κινηματογραφούνται.

Σύμφωνα με την παράγραφο 3 του ίδιου άρθρου: "Όποιος αθέμιτα κάνει χρήση της πληροφορίας ή του υλικού φορέα επί του οποίου αυτή έχει αποτυπωθεί με τους τρόπους που προβλέπονται στις παραγράφους 1 και 2 τιμωρείται με φυλάκιση έως τρία έτη ή χρηματική ποινή". Επιπροσθέτως δηλαδή, η κακόβουλη χρήση αυτών των πληροφοριών που αποκτούνται με τον παραπάνω τρόπο, επιφέρει σχεδόν κακουργηματική ποινή, με στέρηση της προσωπικής ελευθερίας μέχρι και τρία χρόνια.

Το Ευρωπαϊκό Δίκαιο, το οποίο βρίσκεται υψηλότερα στην ιεραρχία από το Εθνικό, έχει θέσει ένα βασικό νομικό πλαίσιο που αφορά όλες τις χώρες της ευρωπαϊκής κοινότητας. Πρόκειται για την τον Κανονισμό 2016/679 του Ευρωπαϊκού Κοινοβουλίου και του Συμβουλίου της 27ης Απριλίου 2016 "για την προστασία των φυσικών προσώπων έναντι της επεξεργασίας των δεδομένων προσωπικού χαρακτήρα και για την ελεύθερη κυκλοφορία των δεδομένων αυτών και την κατάργηση της οδηγίας 95/46/ΕΚ (Γενικός Κανονισμός για την Προστασία Δεδομένων)". Είναι γενικοί κανόνες που, όπως αναφέρει στο άρθρο 2 του Κανονισμού, έχουν πεδίο εφαρμογής στην αυτοματοποιημένη επεξεργασία δεδομένων προσωπικού χαρακτήρα, καθώς και στη μη αυτοματοποιημένη επεξεργασία τέτοιων δεδομένων τα οποία περιλαμβάνονται ή πρόκειται να περιληφθούν σε σύστημα αρχειοθέτησης.

Το ελληνικό δίκαιο υποχρεούται να ακολουθεί μια φιλοσοφία εναρμόνισης με το ευρωπαϊκό, πέρα από την θέσπιση απαγορευτικών κανόνων στον ίδιο τον Ποινικό Κώδικα, κι έτσι δημιούργησε ειδικότερους νόμους σχετικά με αυτό. Η ελληνική νομοθεσία καθορίζει αρχές, δικαιώματα και υποχρεώσεις που αφορούν τη διαχείριση τέτοιων δεδομένων.

Η πιο άμεση μεταφορά της ευρωπαϊκής νομοθεσίας είναι ο Ν. 4624/2019 "Αρχή Προστασίας Δεδομένων Προσωπικού Χαρακτήρα", ο οποίος κυρώθηκε και ισχύει, εξελισσόμενος σταδιακά για να περιγράψει ειδικότερα περιορισμούς του αρχικού Ν. 2472/1997 "Προστασία του ατόμου από την επεξεργασία δεδομένων προσωπικού χαρακτήρα". Σκοπός του είναι η προστασία των φυσικών προσώπων έναντι της επεξεργασίας δεδομένων προσωπικού χαρακτήρα από αρμόδιες αρχές για τους σκοπούς της πρόληψης, διερεύνησης, ανίχνευσης ή δίωξης ποινικών αδικημάτων ή της εκτέλεσης ποινικών κυρώσεων και για την ελεύθερη κυκλοφορία των δεδομένων αυτών.

Ακόμη αξίζει να μνημονευθεί ο Ν. 3917/2011 "Διατήρηση δεδομένων που παράγονται ή υποβάλλονται σε επεξεργασία σε συνάρτηση με την παροχή διαθέσιμων στο κοινό υπηρεσιών ηλεκτρονικών επικοινωνιών ή δημόσιων δικτύων επικοινωνιών, χρήση συστημάτων επιτήρησης μετη λήψη ή καταγραφή ήχου ή εικόνας σε δημόσιους χώρους και συναφείς διατάξεις". Σε αυτόν καθορίζονται μεταξύ άλλων η απόκτηση, διατήρηση και καταστροφή δεδομένων προσωπικού χαρακτήρα. Επίσης, στο Κεφάλαιο Β' αναφέρονται πώς μπορούν να καταγραφούν ήχος και εικόνα σε δημόσιους χώρους με χρήση συστημάτων επιτήρησης.

Σε όλους τους παραπάνω νόμους αναφέρεται με κάποιο τρόπο ότι οι ελληνικές διατάξεις έχουν ως σκοπό την ενσωμάτωση στην εθνική έννομη τάξη οδηγιών ή κανόνων του ευρωπαϊκού κοινοβουλίου.

Πέρα από την κείμενη νομοθεσία για τα δεδομένα, επίσης και η καθαυτή πτήση drone έχει σαφώς καθορισμένους περιορισμούς. Έτσι, απαγορεύεται η πτήση:
α) Πάνω από ανθρώπους και συγκεντρώσεις ανθρώπων.
β) Σε εναέριο χώρο όπου διενεργούνται πτήσεις επανδρωμένων αεροσκαφών.
γ) Εντός των ζωνών κυκλοφορίας των αεροδρομίων σε απόσταση μικρότερη των 8 χιλιομέτρων απο την περίμετρο του αεροδρομίου και από τα ίχνη προσγείωσης/απογείωσης από/προς αεροδρόμιο.
δ) Πάνω, πλησίον και εντός στρατιωτικών περιοχών, εγκαταστάσεων και ανάλογων υποδομών.
ε) Πάνω, πλησίον και εντός στρατιωτικών ή πολιτικών αεροδρομίων και ελικοδρομίων.
στ) Πάνω και πλησίον φυλακών, νοσοκομεία, ιδρύματα, αρχαιολογικούς χώρους και σχολεία.
ζ) Σε περιοχές περιβαλλοντολογικής προστασίας.
η) Σε απαγορευμένες, περιορισμένες, επικίνδυνες και δεσμευμένες περιοχές όπως αυτές αναφέρονται στις αεροναυτικές εκδόσεις (NOTAM).
θ) Τριάντα λεπτά πριν την ανατολή και μισή ώρα μετά τη δύση του ηλίου.

Σημειώνεται ότι επιτρέπεται, υπό προϋποθέσεις, η πραγματοποίηση πτήσεων σε μερικές από τις παραπάνω περιπτώσεις, όπως για παράδειγμα στην περίπτωση (στ) αν γίνεται από στρατιωτικό προσωπικό της εκάστοτε εγκατάστασης κατόπιν ρητής διαταγής και στην περίπτωση (α) από τη δημόσια αρχή κατόπιν εισαγγελικής εντολής για την πρόληψη και αποτροπή έκνομων ενεργειών.

Επίσης, σε περίπτωση που το σύστημα έχει εμβέλεια τηλεχειρισμού μεγαλύτερη από 50 μέτρα απαιτείται καταχώρισή του σε βάση δεδομένων της Πολιτικής Αεροπορίας (Μασχαλίδης, 2018).

ΚΕΦΑΛΑΙΟ 3
RASPBERRY PI

3.1 – ΟΡΙΣΜΟΣ

Το Raspberry Pi είναι το όνομα μιας σειράς υπολογιστών κατασκευασμένους από την Raspberry Pi Foundation που ξεκίνησε από τη Μ. Βρετανία ως Φιλανθρωπικό Ίδρυμα και αποσκοπούσε στην εκπαίδευση των ανθρώπων στον προγραμματισμό και να καταστήσει ευκολότερη την πρόσβαση σε αυτές τις γνώσεις.

Ξεκίνησε την εμπορική διάθεσή του το 2012, με αρκετές παραλλαγές στο ιστορικό τους από τότε. Αν και το πρώτο Raspberry Pi είχε μονοπύρηνο επεξεργαστή στα 700 MHz και με 256 MB μέγεθος κύριας μνήμης, οι παρούσες εκδόσεις αυτών των μικρών υπολογιστών έχουν πολλαπλασιάσει τις δυνατότητές τους και χρησιμοποιούνται πια παγκόσμια.

Μερικές εφαρμογές του είναι η ανάπτυξη προγραμματιστικών δεξιοτήτων, η εφαρμογή σε έργα υλικού και οικιακού αυτοματισμού, η υλοποίηση συστάδων Kubernetes, η Edge προσέγγιση προγραμματισμού, ακόμα και η χρήση σε βιομηχανικές εφαρμογές ή και εξόρυξη κρυπτονομισμάτων. Κάθε νέο μοντέλο συνοδεύεται από την διανομή επίσημων σχεδίων των υποσυστημάτων και των προδιαγραφών του, αλλά δεν πρόκειται για ελεύθερο hardware υπολογιστή και συνεπώς η κατασκευή του γίνεται αποκλειστικά από τη Raspberry Pi Foundation.

Συνήθως τρέχει κάποια έκδοση Linux, με το επίσημο προϊόν λογισμικού του να είναι το Raspberry Pi OS το οποίο είναι βασισμένο στο Debian. Η πρώτη έκδοσή του το Σεπτέμβριο του 2013 ονομαζόταν Raspbian. Υποστηρίζει όμως και άλλα λειτουργικά συστήματα όπως Ubuntu Server, Ubuntu Mate, Twister OS, LAKKA και το RetroPie που αναβιώνει παλιές παιχνιδομηχανές με τη μορφή emulators (Molloy, 2016).

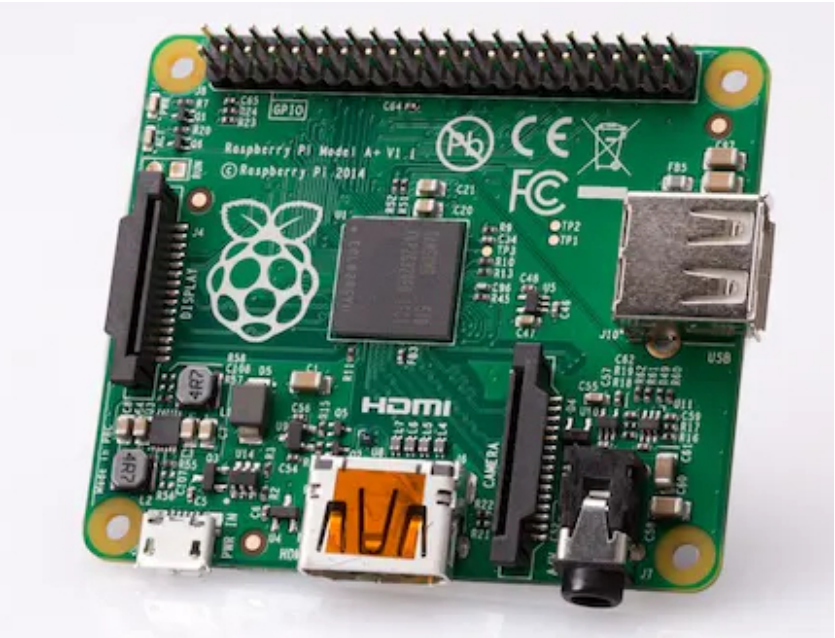
Το Raspberry Pi ανήκει στην κατηγορία των Single Board Computers (SBC). Ένας SBC είναι ολοκληρωμένος υπολογιστής που εκτελεί εντολές εισόδου - εξόδου, περιλαμβάνει μικροεπεξεργαστή, μη επεκτάσιμη μνήμη και ελεγκτές, όλα τοποθετημένα σε μια ενιαία πλακέτα, με υποδοχές για επεκτάσεις περιφερειακών. Παρ' όλο το μέγεθός του και τις περιορισμένες δυνατότητές του, μπορεί να εκκινεί ένα ολοκληρωμένο λειτουργικό σύστημα.

Ενώ οι επιτραπέζιοι υπολογιστές δίνουν έμφαση στην προσαρμοστικότητα που μπορεί να δώσει ο κάθε χρήστης με τις επεκτάσεις ή αντικαταστάσεις των υποσυστημάτων του, η απλουστευμένη και αμετάβλητη δομή ενός SBC έχει ως απόρροια μειωμένα bugs, λιγότερες συγκρούσεις διαλειτουργικότητας και άλλα προβλήματα που προκαλούν διακοπές λειτουργίας. Αυτός είναι και ο λόγος που τους έχει αποδοθεί ως ο "πάντα σε λειτουργία" υπολογιστής που θα αναλάβει καθοδήγηση πυραυλικών συστημάτων, ελεγκτών φωτεινής σήμανσης κυκλοφορίας, συστημάτων αυτόματης πέδησης και ευστάθειας οχημάτων και μηχανημάτων ιατρικής απεικόνισης. Ακόμη, τέτοιοι υπολογιστές απαντώνται σε καταναλωτικά αγαθά που προορίζονται για πιο ευρύ κοινό, όπως κινητά τηλέφωνα και κονσόλες βιντεοπαιχνιδιών. Όλα τα παραπάνω διέπονται από μια αυξημένη αξιοπιστία.

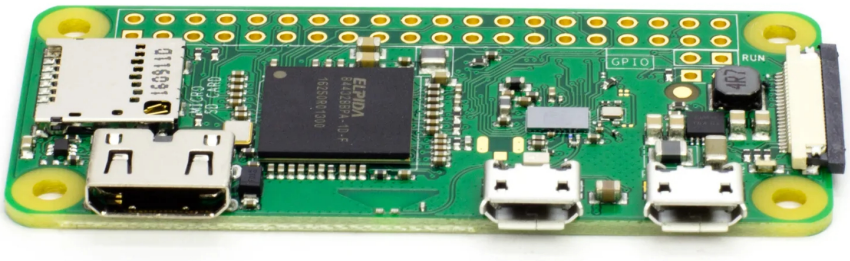
3.2 – ΕΚΔΟΣΕΙΣ RASPBERRY PI

Μέχρι στιγμής έχουν υλοποιηθεί αρκετές εκδόσεις αυτού του Single Board Computer. Σε κάθε γενιά συνήθως υπήρξαν τα model A, με την βελτιωμένη έκδοσή τους τα model B. Τα model A ήταν γενικότερα η φτηνότερη έκδοση με μικρότερη μνήμη και λιγότερες θύρες, όπως USB και Ethernet. Το Raspberry Pi Zero αποτέλεσε ένα ανεξάρτητο παρακλάδι της γραμμής κατασκευής του αρχικού Raspberry Pi 1, με ακόμα μικρότερη πλακέτα και κόστος (Open Source, 2022). Οι εκδόσεις μέχρι στιγμής είναι:

- Pi 1 Model B (2012)
- Pi 1 Model A (2013)
- Pi 1 Model B+ (2014)
- Pi 1 Model A+ (2014)
- Pi 2 Model B (2015)
- Pi Zero (2015)
- Pi 3 Model B (2016)
- Pi Zero W (2017)
- Pi 3 Model B+ (2018)
- Pi 3 Model A+ (2019)
- Pi 4 Model A (2019)
- Pi 4 Model B (2020)
- Pi 400 (2021)
- Pi 5 (2023)



Εικόνα 3.1
Raspberry Pi 1 model A+
https://www.raspberrypi.com/products/raspberry-pi-1-model-a-plus/

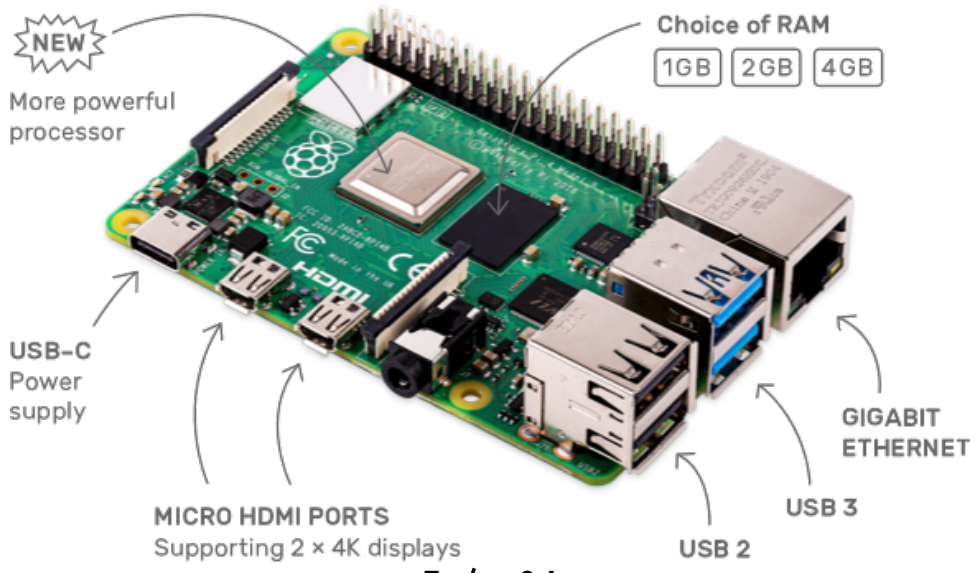


Εικόνα 3.2
Raspberry Pi Zero W
https://www.raspberrypi.com/products/raspberry-pi-zero-w/

Εικόνα 3.3
Raspberry Pi 4 model B
https://nettop.gr/index.php/raspberry-pi/kit-plaketes/raspberry-pi-4-model-b-4gb.html?src=raspberrypi

3.3 - RASPBERRY PI 4 – MODEL B

Στην παρούσα διπλωματική εργασία χρησιμοποιήθηκε αυτό το μοντέλο, με την έκδοση της μνήμης 1 GiB. Οι κατασκευαστικές του προδιαγραφές δίνονται κάτωθι (Doole, 2020).



Εικόνα 3.4
Raspberry Pi 4 model B με τα βασικά χαρακτηριστικά του
https://www.raspberrypi.com/products/raspberry-pi-4-model-b/?variant=raspberry-pi-4-model-b-1gb

Κυριότερες κατασκευαστικές προδιαγραφές

Επεξεργαστής	Broadcom BCM2711, Cortex-A72 (ARM v8 64-bit) με τέσσερις πυρήνες, χρονισμένος στο 1.5 GHz
Νήματα επεξεργαστή	4
Ελεγκτής γραφικών	H.264 (1080p, 60 frames per second (fps) αναπαραγωγή, 1080p, 30 fps εγγραφή), γραφικά OpenGL ES 3.0, H.265 (4K, 60 fps αναπαραγωγή)
Μνήμη	1 GiB ή 2 GiB ή 4 GiB LPDDR4
Τροφοδοσία	5 Volt (ελάχιστο 3 Volt), υποδοχή USB Type C
GPIO	40 ακίδες
LAN	Διαθέσιμο
PoE (Power Over Ethernet)[1]	Ενεργοποιημένο
WiFi	Διαθέσιμο, 2.4 GHz και 5.0 GHz IEEE 802.11b/g/n/ac
Bluetooth	5.0
Κάρτα SD	Διαθέτει
HDMI	2 υποδοχές Mini HDMI με 4K ανάλυση
USB	Δύο υποδοχές USB 2.0, δύο υποδοχές USB 3.0
Κάμερα	Σειριακή διεπαφή (CSI – Camera Serial Interface)
Ήχος	3.5 mm jack στερεοφωνικού ήχου
Απεικόνιση	Σειριακή διεπαφή (DSI – Display Serial Interface)
Λειτουργία	0^0C έως 50^0C

3.4 - ΕΠΕΞΕΡΓΑΣΤΗΣ ARM

Το Raspberry Pi είναι εξοπλισμένο με επεξεργαστή αρχιτεκτονικής ARM της οποίας το ακρωνύμιο σημαίνει Advanced RISC Machine. Ο πρώτος επεξεργαστής ARM παράχθηκε το 1978 στο Πανεπιστήμιο του Cambridge, ενώ ο πρώτος επεξεργαστής ARM RISC το 1985 από την εταιρεία Acorn Group of Computers. Διαδόθηκαν και χρησιμοποιήθηκαν ευρέως σε φορητές συσκευές όπως ψηφιακές κάμερες, έξυπνα κινητά τηλέφωνα, οικιακές μονάδες δικτύωσης και ασύρματης επικοινωνίας και άλλα ενσωματωμένα συστήματα λόγω των πλεονεκτημάτων τους, που αποφέρουν αποδεκτά επίπεδα απόδοσης με χαμηλές απαιτήσεις ενέργειας.

Οι επεξεργαστές RISC (Reduced Instruction Set Computer) διαθέτουν μειωμένο σετ εντολών, οι οποίες είναι και αυτές που χρησιμοποιούνται συχνότερα. Σκοπός της δημιουργίας τους είναι ένας ταχύτερος και παράλληλα οικονομικότερος επεξεργαστής για χρήση σε απλούστερους υπολογισμούς. Όταν απαιτούνται πιο σύνθετοι υπολογισμοί, χρησιμοποιούνται επεξεργαστές CISC (Reduced Instruction Set Computer) με πλήρες σετ εντολών οι οποίοι κατά κανόνα είναι ακριβότεροι.

Χαρακτηριστικό παράδειγμα της διαφοράς των δύο αυτών αρχιτεκτονικών είναι ο πολλαπλασιασμός δύο αριθμών, σε χαμηλό επίπεδο υλικού αναπαριστώμενο με Assembly.

Βασικός στόχος της CISC αρχιτεκτονικής είναι να διεκπεραιώσει την ενέργεια με όσο το δυνατόν λιγότερες γραμμές Assembly. Έτσι οι επεξεργαστές αυτοί έχουν ενσωματώσει την

1 Το Power Over Ethernet είναι μια τεχνολογία που επιτρέπει την διοχέτευση ρεύματος με το καλώδιο Ethernet. Αυτό σημαίνει ότι για παράδειγμα, μια κάμερα συνδεδεμένη με το Raspberry Pi Μέσω καλωδίου Ethernet μπορεί και να τροφοδοτηθεί με ρεύμα για να λειτουργήσει χωρίς εξωτερική παροχή. Απαιτείται ξεχωριστό HAT.

εντολή MULT, η οποία θεωρείται σύνθετη. Με αυτή τη μία εντολή ο επεξεργαστής λαμβάνει τα περιεχόμενα από δύο θέσεις της κύριας μνήμης (οι παράγοντες του πολλαπλασιασμού) και τις αποθηκεύει σε δύο καταχωρητές του επεξεργαστή. Πολλαπλασιάζει τα περιεχόμενα των καταχωρητών στη Μονάδα Εκτέλεσης (Execution Unit) και αποθηκεύει το γινόμενο σε έναν άλλο καταχωρητή. Έτσι, ο πολλαπλασιασμός γίνεται με μία και μοναδική εντολή Assembly:

```
MULT address_1, address_2
```

Επιδρά ουσιαστικά απευθείας στην κύρια μνήμη και θυμίζει αρκετά εντολή γλώσσας προγραμματισμού υψηλού επιπέδου. Ένα άμεσο πλεονέκτημα αυτής της προσέγγισης είναι ότι ο μεταγλωττιστής μεταφράζει εύκολα μια εντολή γλώσσας προγραμματισμού υψηλού επιπέδου σε Assembly. Επίσης, επειδή ο κώδικας έχει μικρότερη έκταση, απαιτείται λιγότερη μνήμη για την αποθήκευση των εντολών. Δίνεται δηλαδή έμφαση στην ενσωμάτωση των εντολών απευθείας πάνω στο υλικό.

Οι επεξεργαστές RISC χρησιμοποιούν πιο απλές εντολές που μπορούν να εκτελεστούν σε ένα κύκλο μηχανής. Έτσι, η προηγούμενη εντολή MULT διαχωρίζεται σε τρεις ξεχωριστές εντολές. Την LOAD που μετακινεί δεδομένα από την κύρια μνήμη σε καταχωρητή, την PROD που πολλαπλασιάζει τα περιεχόμενα καταχωρητών και την STORE που μετακινεί δεδομένα από καταχωρητή στην κύρια μνήμη. Εδώ θα χρειάζονταν τέσσερις γραμμές Assembly (Chen, et al., 2000):

```
LOAD A, address_1
LOAD B, address_2
PROD A, B
STORE address_1, A
```

Η προσέγγιση αυτή έχει τα δικά της πλεονεκτήματα. Οι εντολές RISC απαιτούν λιγότερα τρανζίστορ αφήνοντας χώρο για περισσότερους καταχωρητές γενικού σκοπού. Επίσης, επειδή κάθε ξεχωριστή εντολή απαιτεί έναν κύκλο, είναι δυνατή η *σωλήνωση (pipelining)*, δηλαδή η έναρξη μιας νέας εντολής πριν τελειώσει η προηγούμενη, άρα και η *παραλληλοποίηση* εργασιών. Ακόμη, μετά την επεξεργασία των δεδομένων των καταχωρητών οι RISC επεξεργαστές διατηρούν το περιεχόμενό τους δυνητικά εξοικονομώντας χρόνο σε περίπτωση που τα περιεχόμενα ξαναχρειαστούν σε άλλη εντολή, ενώ οι CISC επεξεργαστές το σβήνουν και απαιτείται η επανάκτησή τους από την κύρια μνήμη.

3.5 – THERMAL THROTTLING

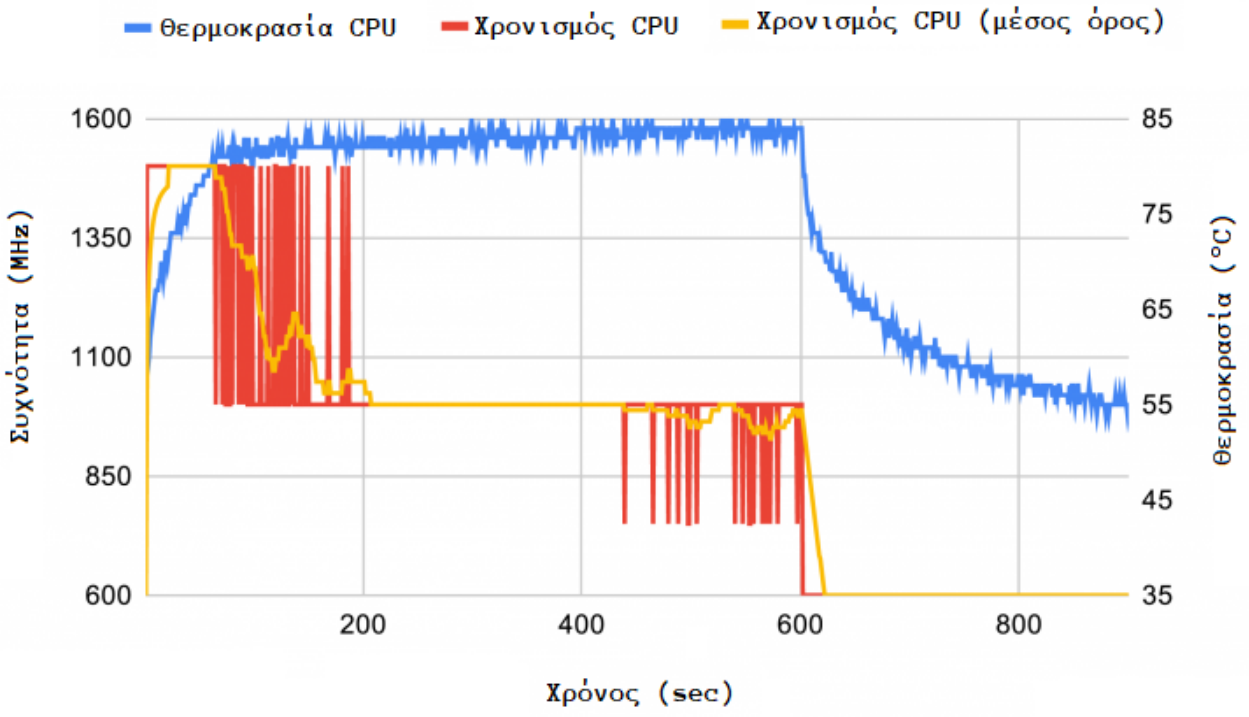
Ο επεξεργαστής του Raspberry Pi λειτουργεί ιδανικά στο παραπάνω αναφερόμενο εύρος θερμοκρασιών από 0^{o}C έως 50^{o}C χωρίς να υποστεί ζημιά το υλικό κατασκευής του. Συνήθως όμως επιτελεί αρκετά βαριές εργασίες με αποτέλεσμα να θερμαίνεται εύκολα. Για την από κατασκευής μείωση της θερμοκρασίας του έχει κάλυμμα κατασκευασμένο από κράμα μετάλλου που απάγει τη θερμότητα από αυτόν.

Θεωρείται δε απαραίτητη η τοποθέτηση heat sink δηλαδή πρόσθετων ψυκτρών απαγωγής θερμότητας πάνω σε επεξεργαστή, μνήμη και ελεγκτή γραφικών, υποσυστήματα στα οποία παρατηρείται το ίδιο φαινόμενο. Επιπροσθέτως, στην αγορά κυκλοφορούν θήκες με ενσωματωμένο ανεμιστήρα που κυκλοφορούν κρύο αέρα που περνάει μέσα από αυτές τις ψύκτρες για την περαιτέρω μείωση της θερμοκρασίας.

Εικόνα 3.5
Ψύκτρες απαγωγής θερμότητας Raspberry Pi (heat sink)
https://www.indiamart.com/proddetail/raspberry-pi-3-in-1-heat-sink-aluminium-22808170688.html

Όμως και το ίδιο το Raspberry Pi έχει ενσωματωμένο firmware το οποίο ενεργοποιεί υπό συνθήκες μια λειτουργία που ονομάζεται *thermal throttling*. Ο επεξεργαστής διαθέτει θερμομετρικό αισθητήρα που αναφέρει τη θερμοκρασία του σε πραγματικό χρόνο. Αν αυτή η θερμοκρασία ξεπεράσει μια τιμή (οριοθετημένη στο firmware), ο επεξεργαστής μειώνει εκούσια την απόδοσή του (την ταχύτητα χρονισμού των πυρήνων του) ώστε η θερμοκρασία να παραμείνει είτε σταθερή είτε, ιδανικά, να μειωθεί σε επιτρεπτά επίπεδα. Η λειτουργία thermal throttling έχει επίπεδα έντασης λειτουργίας, με άλλα λόγια όσο πιο δραματικά αυξάνεται η θερμοκρασία τόσο περισσότερο θα μειώνεται η απόδοση του επεξεργαστή (Bate, 2019).

Αυτός είναι ο λόγος που η επαλήθευση ορθής λειτουργίας ορισμένων υλοποιήσεων (ειδικά των απαιτητικών) εφαρμογών δοκιμάζεται και σε εκτεταμένο διάστημα χρόνου, ώστε να ελεχθεί αν η ενεργοποίηση του thermal throttling επηρεάσει την όλη λειτουργία.



Εικόνα 3.6
Φαινόμενο thermal throttling
https://www.raspberrypi.com/news/thermal-testing-raspberry-pi-4/

Η παραπάνω εικόνα δείχνει το φαινόμενο thermal throttling σε διάστημα χρόνου 800 δευτερολέπτων. Στην αρχή του χρόνου, ο επεξεργαστής πιέζεται με βαρύ φόρτο εργασίας και φθάνει στα 1500 MHz, όσο και είναι η μέγιστη συχνότητά του. Η θερμοκρασία του ανεβαίνει

στους 85^{0}C, οπότε, μετά από τα 200 δευτερόλεπτα η συχνότητα πέφτει εκούσια περίπου στα 1000 MHz. Επειδή αυτό δεν είναι αρκετό για να μειωθεί η θερμοκρασία, η συχνότητα πέφτει ακόμα περισσότερο μετά τα 400 δευτερόλεπτα, ως τα 600 MHz. Εκεί η θερμοκρασία αρχίζει να πέφτει ώστε να φθάσει περίπου στους 55^{0}C, όπου είναι και η εργοστασιακή τιμή ασφαλείας. Οι μετρήσεις πραγματοποιήθηκαν από την ίδια την Raspberry Pi Foundation και ανακοινώθηκαν την πρώτη μέρα κυκλοφορίας του Raspberry Pi 4 στην αγορά.

3.6 – GPIO PINS

Πάνω στην πλακέτα του Raspberry Pi 4 (model B) υπάρχουν 40 ακίδες. Ονομάζονται GPIO Pins, δηλαδή ακίδες γενικής χρήσης εισόδου – εξόδου και επιτρέπουν τον έλεγχο ηλεκτρονικών εξαρτημάτων. Αυτές μπορούν να συνδεθούν με μικρά καλώδια σε breadboard και μέσω αυτού σε άλλες συσκευές. Υπάρχουν διάφορες βιβλιοθήκες της Python για τον απευθείας προγραμματισμό αυτών των ακίδων με scripts ή snippets κώδικα (Eddison, 2018).

Επίσης, οι ακίδες έχουν συγκεκριμένη απόσταση μεταξύ τους. Έχουν κατασκευαστεί πλακέτες, από τον επίσημο κατασκευαστή του Raspberry Pi αλλά και από τρίτους, που φέρουν υποδοχές ίσες με αυτές τις αποστάσεις ώστε να συνδεθούν με το Raspberry Pi. Αυτές οι πλακέτες ονομάζονται Hardware Attached on Top, με το ακρωνύμιο HAT ως λογοπαίγνιο, επειδή τοποθετούνται πάνω από την κύρια πλακέτα, σαν "καπέλο", αφήνοντας ένα κενό 8 χιλιοστών ώστε να μην δημιουργήσουν ζημιά σ' αυτή. Είναι κατασκευασμένες για διάφορες λειτουργίες, όπως μέτρηση θερμοκρασίας, βαρομετρικής πίεσης, διατάξεις λαμπτήρων LED, κάρτες ήχου και άλλα (Adams, 2014).

Κάθε HAT έχει έγκυρο EEPROM ID στο οποίο περιέχονται πληροφορίες σχετικά με τον προμηθευτή, το χάρτη διασύνδεσης GPIO με τη λειτουργία της κάθε οπής και τη δενδρική δομή της συσκευής. Το EEPROM είναι ακρωνύμιο για Electrically Erasable Programmable Read Only Memory και αποθηκεύει ένα μικρό κομμάτι επανεγγράψιμων δεδομένων πάνω στο υλικό. Η δενδρική δομή είναι ουσιαστικά η περιγραφή αυτού του υλικού και επιτρέπει την απευθείας φόρτωση των drivers στο λειτουργικό σύστημα.

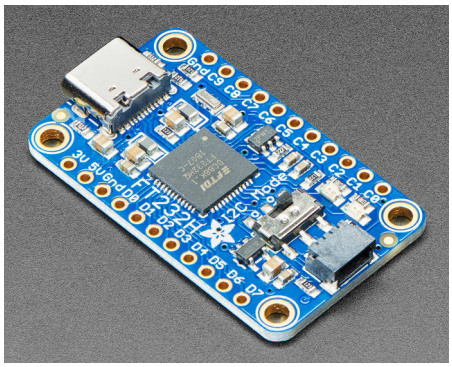
Με την προσάρτηση μιας τέτοιας πλακέτας είναι αναγκαίοι και οι αντίστοιχοι drivers (προγράμματα οδήγησης) ώστε να εκκινούνται οι λειτουργίες μαζί με το σύστημα ή χειρωνακτικά μέσω του τερματικού. Το Raspberry Pi δε γνωρίζει αν υπάρχει συνδεδεμένη πλακέτα, ενώ οι οδηγοί όταν φορτώνονται υποθέτουν ότι έχουν στην αποκλειστική διάθεσή τους την διεπαφή GPIO. Λόγω των διαφορετικών εκδόσεων των Raspberry Pi και των αντίστοιχων drivers που είναι διαθέσιμοι, ορισμένες φορές δημιουργείται σύγχυση διότι οι τελευταίοι είναι γραμμένοι ώστε να θεωρούν απόλυτα διαθέσιμες συγκεκριμένες ακίδες (Khan, 2022). Για παράδειγμα, υπάρχουν Raspberry Pi με σετ των 26 και με σετ των 40 ακίδων. Για την πιο εύκολη backwards compatibility (οπισθόδρομη συμβατότητα), οι λειτουργίες των πρώτων 26 ακίδων δεν έχουν μεταβληθεί στις εκδόσεις με τα σετ των 40 ακίδων.

3.7 – ΑΛΛΟΙ SINGLE BOARD COMPUTERS

Πέρα από το Raspberry Pi, αξιομνημόνευτες περιπτώσεις Single Board Computers που όμως δεν τρέχουν ολοκληρωμένο λειτουργικό σύστημα είναι το Adafruit και το Arduino. Παρ' όλη την απλότητά τους, παρέχουν διάφορα εργαλεία για την υλοποίηση projects και η φιλοσοφία τους βασίζεται στην αποθήκευση σε αυτά κώδικα που τρέχει σε ατέρμονα βρόχο με την εκκίνηση του συστήματος. Πρόκειται δηλαδή για μια κατηγορία από προγραμματιζόμενα *ενσωματωμένα συστήματα*.

Εικόνα 3.7
Λογότυπο της Adafruit
https://swling.com/blog/wp-content/uploads/2015/10/Adafruit_logo-3.png

Η Adafruit Industries είναι μια εταιρεία κατασκευής υλικού υπολογιστών, εργαλείων και εξαρτημάτων, που ιδρύθηκε το 2005 από τη Limor Fried και έχει έδρα τη Νέα Υόρκη. Είναι επίσης ο κύριος προμηθευτής επιπρόσθετων πλακετών συμβατών με Raspberry Pi.



Εικόνα 3.8
Adafruit FT232H, γενικής χρήσης module, από USB σε GPIO+SPI+I2C
https://www.adafruit.com/product/2264

Η Arduino είναι μια εταιρεία που κατασκευάζει hardware (και αντίστοιχο software) Single Board Microcontrollers (Μικροελεγκτές Μονής Πλακέτας) και kit κατασκευής ψηφιακών συσκευών. Σημειώνεται ότι τα προϊόντα αυτά καλύπτονται από την GNU General Publc Licence συνεπώς επιτρέπεται η κατασκευή τους από οποιονδήποτε πέρα από την ίδια την εταιρεία.

Εικόνα 3.9
Λογότυπο της Arduino
https://logodownload.org/wp-content/uploads/2019/03/arduino-logo-0.png

Οι microcontrollers αυτοί προγραμματίζονται με παραλλαγές των C και C++, με ένα εξειδικευμένο προγραμματιστικό interface (διεπαφή) που ονομάζεται γλώσσα Arduino. Σε αυτό γράφεται ο κώδικας που τρέχει συνεχώς όταν ο ελεγκτής τροφοδοτείται με ρεύμα.

Εικόνα 3.10
Arduino Uno r3
https://www.why.gr/%CE%BA%CE%B1%CF%84%CE%B1%CF%83%CF%84%CE%B7%CE%BC%CE%B1/open-hardware/arduino/arduino-main-boards/arduino-uno-r3/

ΚΕΦΑΛΑΙΟ 4

ΚΡΥΠΤΟΓΡΑΦΙΑ

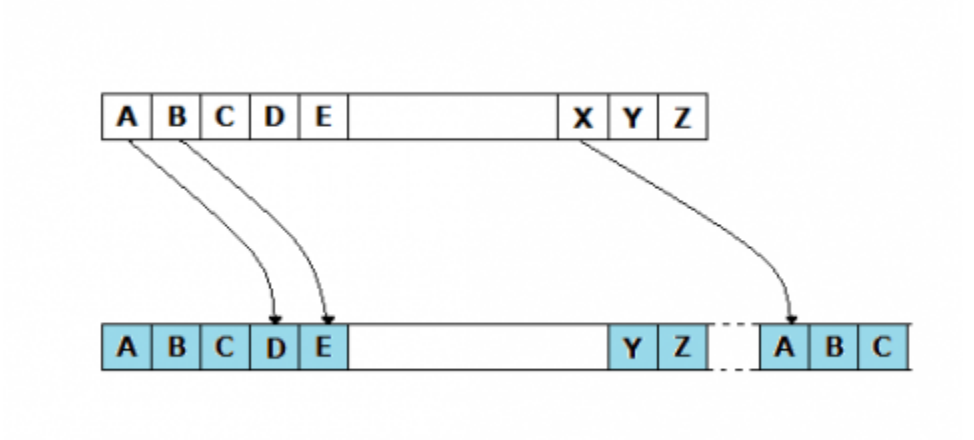
4.1 – ΙΣΤΟΡΙΚΗ ΑΝΑΔΡΟΜΗ

Η Κρυπτολογία είναι μια νέα επιστήμη. Αν και οι πρώιμες τεχνικές της άρχισαν να χρησιμοποιούνται χιλιάδες χρόνια πριν για την απόκρυψη μυστικών μηνυμάτων, η συστηματική μελέτη της ως επιστήμη ξεκίνησε περίπου εκατό χρόνια πριν.

Η πρώτη καταγεγραμμένη απόδειξη χρήσης κρυπτογραφίας με κάποια μορφή είναι σε μια επιγραφή που βρέθηκε στον τάφο του ευγενή Knumhotep II της αρχαίας Αιγύπτου και χρονολογείται περίπου στο 1900 πΧ. Ο γραφέας χρησιμοποίησε σε αρκετά σημεία ασυνήθιστα ιερογλυφικά στη θέση των πιο κοινών. Σκοπός του δεν ήταν να αποκρύψει το μήνυμα αλλά να το κάνει να φαίνεται πιο αξιοπρεπές κι επίσημο. Η γραφή δεν αποτελούσε μορφή μυστικής γραφής μεν, αλλά υλοποίησε μια μορφή παραλλαγής του αρχικού κειμένου, και είναι η πρώτη εμφάνισης τέτοιας περίπτωσης.

Ενδείξεις μορφών κρυπτογραφίας είναι διάσπαρτες στους επιφανέστερους αρχαίους πολιτισμούς. Το “Arthshashtra” (Η Επιστήμη των Υλικών Αποδοχών), ένας τόμος καταγραφής κρατικών υποθέσεων της Ινδίας που έγραψε ο φιλόσοφος Kautalya περίπου το 300 πΧ, περιέχει μεταξύ άλλων και αναθέσεις αποστολών σε Ινδούς κατασκόπους με “μυστική γραφή” (Singh, 2000).

Η πιο γνωστή ίσως περίπτωση κρυπτογράφησης είναι αυτή του Ιουλίου Καίσαρα που χρονολογείται περίπου στο 100 πΧ. Ο αλγόριθμος είναι γνωστός ως Αλγόριθμος του Καίσαρα και υλοποιεί μια σταθερή μονοαλφαβητική αντικατάσταση στα γράμματα του λατινικού αλφαβήτου. Για παράδειγμα με το κλειδί 3, κάθε γράμμα του αρχικού κειμένου αντικαθίστατο με αυτό που ήταν 3 γράμματα μετά στο λατινικό αλφάβητο. Αυτό σημαίνει ότι μια επαναφορά κάθε γράμματος του κρυπτομηνύματος 3 θέσεις πίσω στο λατινικό αλφάβητο θα αποκάλυπτε το αρχικό μήνυμα.



Εικόνα 4.1
Ο Αλγόριθμος του Καίσαρα
https://www.redhat.com/rhdc/managed-files/caeser-cipher_1.png

Μια ενδιαφέρουσα τεχνική χρησιμοποίησαν οι αρχαίοι Σπαρτιάτες. Τύλιγαν μια λωρίδα δέρματος σε μια ράβδο συγκεκριμένου πάχους, που λεγόταν *σκυτάλη* ή *κρυπτεία*, και μετά έγραφαν πάνω της το μήνυμα προς αποστολή. Η λωρίδα, χωρίς τη σκυτάλη, παραδινόταν στον αγγελιοφόρο. Αν η λωρίδα έπεφτε σε λάθος χέρια στη διαδρομή, το μήνυμα δε θα μπορούσε

να διαβαστεί απευθείας (ούτε καν από τον ίδιο τον αγγελιοφόρο). Μπορούσε να το διαβάσει ο τελικός παραλήπτης, που τύλιγε μυστικά τη λωρίδα σε μια αντίστοιχη σκυτάλη γνωρίζοντας το πάχος της αρχικής. Μιας το κλειδί τόσο της κρυπτογράφησης όσο και της αποκρυπτογράφησης (η σκυτάλη) ήταν το ίδιο, μπορούμε να πούμε ότι πρόκειται για ένα είδος συμμετρικού αλγόριθμου (Sidhpurwala, 2013). Ένα ακόμη σημαντικό στοιχείο είναι ότι μπορούσε να επαληθευτεί η ταυτότητα του αποστολέα και του μηνύματος, ανάλογα με το αν το τελικό μήνυμα μπορούσε να διαβαστεί ή όχι, κάπως ανάλογο με τις σημερινές hash functions (συναρτήσεις κατακερματισμού). Από αυτή τη διαδικασία σήμερα σώζονται οι σκυταλοδρομίες ως άθλημα, οι οποίες δεν ήταν άλλο παρά ασκήσεις προπόνησης για τους δρομείς ώστε να διατηρούν καλούς τους χρόνους παράδοσης των μηνυμάτων.

Εικόνα 4.2
Η Σπαρτιατική Σκυτάλη
https://defensegr.files.wordpress.com/2020/04/h_kryptografia_ths_skytalhs_07.jpg

Εύκολα συμπεραίνεται ότι στις διαδικασίες που ακολουθούνταν στην αρχαιότητα, η ασφάλεια του συστήματος βασιζόταν στη μυστικότητα του αλγορίθμου και όχι στου μυστικού κλειδιού. Με τη γνώση της διαδικασίας το αρχικό μήνυμα μπορούσε να ανακτηθεί εύκολα. Συγκεκριμένα στους αλγόριθμους αντικατάστασης, το κρυπτομήνυμα μπορούσε να αναλυθεί με την παρατήρηση της συχνότητας εμφάνισης των χαρακτήρων σ' αυτό.

Κομβικό σημείο στην εξέλιξη των αλγορίθμων κρυπτογράφησης ήταν το κρυπτοσύστημα Vernam-Vigenere. Επινοήθηκε το 1553 από τον Ιταλό Giovan Battista Bellaso όμως για αιώνες αποδιδόταν στο Γάλλο κρυπτογράφο του 16ου αιώνα Blaise de Vigenere που δημιούργησε ανεξάρτητα ένα παρόμοιο του 1586. Για πολλά χρόνια θεωρούνταν απρόσβλητος και μάλιστα πήρε το όνομα "le chiffre indechiffrable" (ο άτρωτος κρυπταλγόριθμος). Είναι η πρώτη επίσημα καταγεγραμμένη περίπτωση όπου η ασφάλεια του κρυπτοσυστήματος βασίζεται στη μυστικότητα *του κλειδιού* και όχι του αλγορίθμου.

Πρόκειται για έναν αλγόριθμο αντικατάστασης στον οποίο κάθε χαρακτήρας του αρχικού μηνύματος κρυπτογραφείται, με διαφορετικό κάθε φορά κλειδί, σε χαρακτήρα του κρυπτοκειμένου. Μια λέξη κλειδί είναι αυτή που υποδεικνύει ποια συγκεκριμένη αντικατάσταση θα πραγματοποιηθεί για κάθε χαρακτήρα. Πρόκειται δηλαδή για μια πολυαλφαβητική αντικατάσταση.

Η λέξη κλειδί διατρέχει, παράλληλα επαναλαμβανόμενη, το αρχικό κείμενο. Κάθε γράμμα της λέξης κλειδιού αντιστοιχεί σε κάποιο γράμμα του αρχικού κειμένου. Με αυτό τον τρόπο, κάθε γράμμα κρυπτογραφείται σε διαφορετικό χαρακτήρα. Έτσι δε μπορεί να κρυπταναλυθεί με παρατήρηση της συχνότητας εμφάνισης των γραμμάτων.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
T	**H**	**I**	**S**	**I**	**S**	**A**	**C**	**I**	**P**	**H**	**E**	**R**	**M**	**E**	**S**	**S**	**A**	**G**	**E**
S	E	C	R	E	T	S	E	C	R	E	T	S	E	C	R	E	T	S	E
L	**L**	**K**	**J**	**M**	**L**	**S**	**G**	**K**	**G**	**L**	**X**	**J**	**Q**	**G**	**J**	**W**	**T**	**Y**	**I**

Εικόνα 4.3
Παράδειγμα αλγόριθμου Vigenere

Στο παραπάνω παράδειγμα, η συμβολοσειρά “THISISACIPHERMESSAGE” με τη λέξη κλειδί “SECRET” κρυπτογραφείται σε “LLKJMLSGKGLXJQGJWTYI”. Η λέξη κλειδί διατρέχει επαναλαμβανόμενη το Αρχικό Κείμενο, μέχρι να τελειώσει και το τελευταίος χαρακτήρας του, ενώ διακόπτεται λίγο μετά την τρίτη επανάληψη. Παρατηρείται ότι το Αρχικό Κείμενο δεν έχει κενά διαστήματα. Αν προστεθούν, τότε με Αρχικό Κείμενο “THIS IS A CIPHER MESSAGE” και την ίδια λέξη κλειδί, το κρυπτοκείμενο που προκύπτει είναι: “KLKIdAJdCqGAGLGHdEWWURKX” που είναι τελείως διαφορετικό από αυτό χωρίς κενά (Sweigart, 2018).

4.2 - ΠΕΡΙ ΚΡΥΠΤΟΓΡΑΦΗΣΗΣ

Στις επιθέσεις αποκρυπτογράφησης θεωρούμε ότι είναι γνωστός στον κρυπταναλυτή ο αλγόριθμος κρυπτογράφησης. Στη μέτρηση δύναμης ενός κρυπτοσυστήματος, το βασικό χαρακτηριστικό αξιοπιστίας του είναι η *αρχή του Kerchoff*: "Η ασφάλεια ενός κρυπτοσυστήματος δεν έγκειται στη μυστικότητα του αλγορίθμου κρυπτογράφησης, αλλά του κλειδιού" (Κάτος, Στεφανίδης, 2003).

Δε γίνεται προσπάθεια απόκρυψης του αλγόριθμου κρυπτογράφησης για σοβαρούς λόγους. Αφενός μεν η αποτίμηση ασφάλειας άγνωστου αλγόριθμου δε μπορεί να είναι αντικειμενική μιας και διαφορετικοί φορείς κρίνουν διαφορετικά ένα αποτέλεσμα χωρίς γνώση του τρόπου υλοποίησης. Αφετέρου δε, μια εξαντλητική έρευνα των λειτουργιών που διαδραματίζονται σε μια κρυπτογράφηση θα μπορούσε να οδηγήσει συμπερασματικά στην ανακάλυψη του αλγόριθμου. Επίσης, αν υπάρξει διαρροή πληροφορίας, το πιο εύκολο στοιχείο να αντικατασταθεί είναι το κλειδί κι όχι ολόκληρος ο αλγόριθμος.

Ένας ορθά δομημένος αλγόριθμος κρυπτογράφησης χαρακτηρίζεται από μέτρα που διατύπωσε ο Claude Shannon το 1949.

4.2.1 - ΜΕΤΡΑ ΤΟΥ SHANNON

1. Βαθμός απαιτούμενης κρυπτογραφικής ασφάλειας.

Αφορά την αποκάλυψη πληροφορίας στον αντίπαλο όταν αυτός επιχειρεί αποκρυπτογράφηση.

2. Μήκος κλειδιού.

Το μήκος κλειδιού επηρεάζει την ευκολία χειρισμού του.

3. Κρυπτογράφηση και αποκρυπτογράφηση στην πράξη.

Η προσπάθεια που απαιτείται πρακτικά στην κρυπτανάλυση (χρόνος ή και πόροι).

4. Διόγκωση κρυπτοκειμένου.

Το κρυπτοκείμενο πρέπει να έχει σχεδόν το ίδιο (ή συγκρίσιμο) μέγεθος με το αρχικό κείμενο.

5. Διάδοση σφαλμάτων κρυπτογράφησης.

Το τυχόν σφάλμα κατά τη διάρκεια κρυπτογράφησης πρέπει να επηρεάζει όσο δυνατόν λιγότερο το κρυπτοκείμενο.

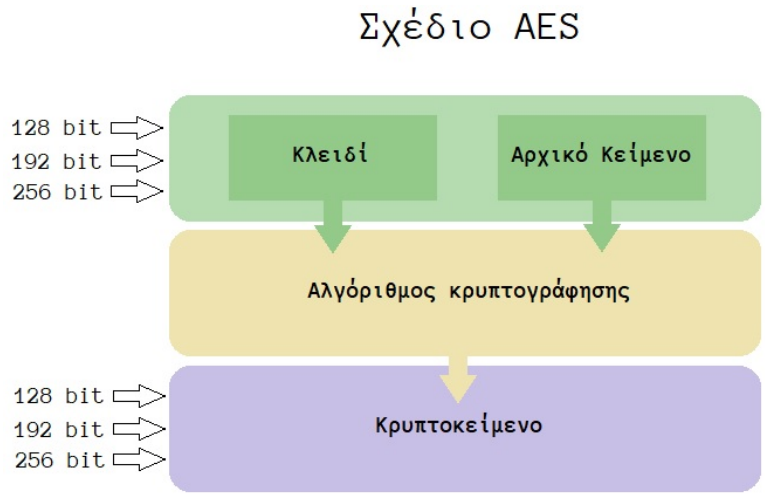
Σύμφωνα πάντα με τον Shannon, υπάρχουν δύο αρχές που αξιολογούν την κρυπτογραφική δύναμη ενός αλγορίθμου.

Σύγχυση: Η ιδιότητα του αλγορίθμου να αποτρέπει τον επιτιθέμενο από την πρόβλεψη του κρυπτοκειμένου με οποιαδήποτε αλλαγή του απλού κειμένου. Ένας κρυπταλγόριθμος προσφέρει υψηλή σύγχυση όταν η σχέση του απλού κειμένου με το κρυπτοκείμενο είναι πολύ πολύπλοκη ώστε να την ανακαλύψει ο αντίπαλος.

Διάχυση: Το χαρακτηριστικό του κρυπταλγόριθμου βάσει του οποίου κάποιο τμήμα του απλού κειμένου, ανεξάρτητα από το μέγεθος και τη θέση του, επηρεάζει όσο μεγαλύτερο τμήμα του κρυπτοκειμένου είναι δυνατόν.

4.3 – ΑΛΓΟΡΙΘΜΟΣ AES (Advanced Encryption System)

Το 1997, το Ινστιτούτο Προτύπων και Τεχνολογίας των ΗΠΑ (National Institute of Standards and Technology - NIST) ξεκίνησε τη διαδικασία εγκαθίδρυσης ενός νέου κρυπτοσυστήματος. Το νέο σύστημα θα ερχόταν να αντικαταστήσει τον αλγόριθμο DES (Data Encryption Standard) που χρησιμοποιούνταν από το 1973. Ο DES δεχόταν επιθέσεις αποκρυπτογράφησης που άρχιζαν να αποδίδουν σε ωφέλιμο χρόνο. Με άλλα λόγια, μπορεί η κρυπτανάλυση δεδομένων που είχαν κρυπτογραφηθεί με τον DES να ήταν ακόμα σχετικά χρονοβόρα, όμως όταν ολοκληρωνόταν οι πληροφορίες ήταν ακόμα αξιοποιήσιμες. Επίσης ο DES ήταν κατασκευασμένος προ εικοσαετίας και απευθυνόταν στο υλικό της εποχής του, πλέον σχετικά παρωχημένο κι ακόμη, δεν ανακοινώθηκαν ποτέ οι προδιαγραφές κατασκευής του κι έτσι ήταν αδύνατες οι μετρικές ως προς την ασφάλειά του.



Εικόνα 4.4
Γενική αναπαράσταση του AES

Για τον AES κατατέθηκαν δεκαπέντε ολοκληρωμένες αλγοριθμικές προτάσεις. Αυτή που τελικά επιλέχθηκε ήταν ο Rijndael των Βέλγων Rijmen και Daemen, που είναι πλέον γνωστό ως αλγόριθμος κρυπτογράφησης AES.

Ο αλγόριθμος AES έχει πέντε τρόπους λειτουργίας:

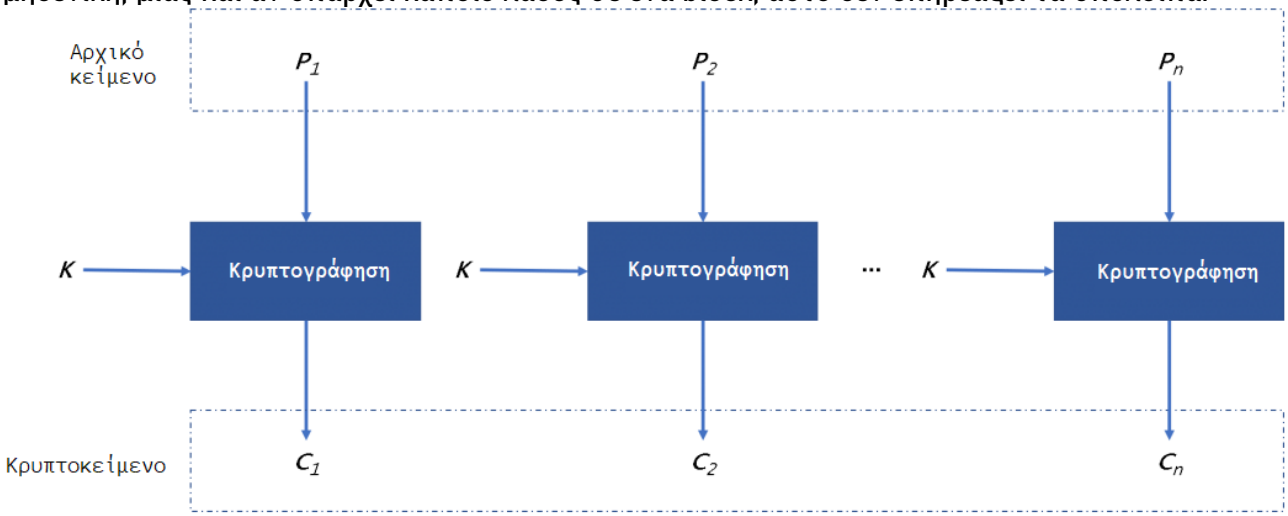
1. ECB: Electronic Code Book
2. CBC: Cipher Block Chaining

3. CFB: Cipher FeedBack
4. OFB: Output FeedBack
5. CTR: Counter

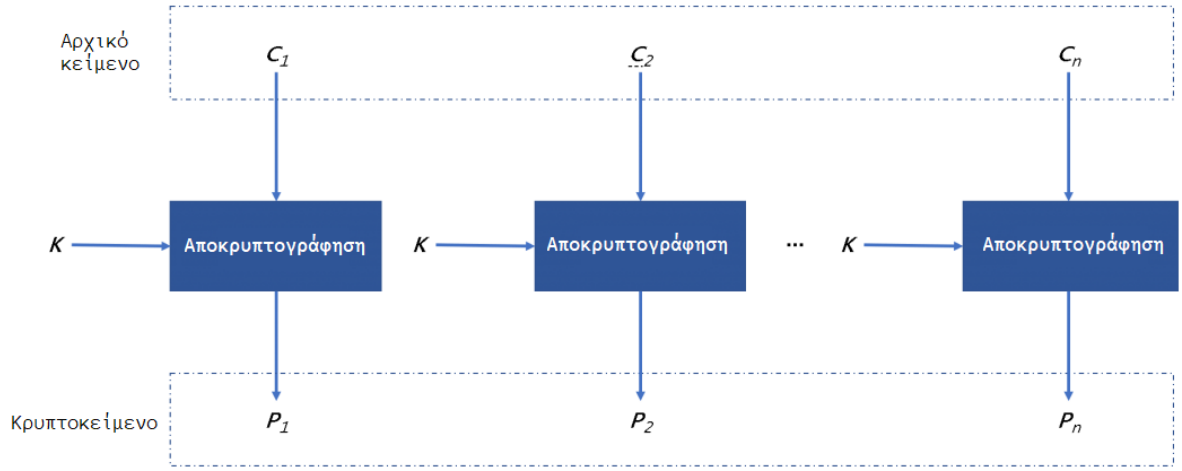
4.3.1 - ECB: Electronic Code Book

Ο ECB είναι ο απλούστερος τρόπος. Δεν προτείνεται γενικά λόγω της απλότητάς του, και παρουσιάζει εμφανείς αδυναμίες. Το αρχικό κείμενο χωρίζεται σε κομμάτια (block) όσο και το μέγεθος του AES (128, 192 ή 256 bits). Ο αλγόριθμος συμπληρώνει το κάθε κομμάτι δεδομένων μέχρι να αποκτήσει το συγκεκριμένο μήκος. Κάθε block κρυπτογραφείται με το ίδιο κλειδί και τον ίδιο αλγόριθμο, οπότε ίδια blocks παράγουν το ίδιο κρυπτοκείμενο. Επιπροσθέτως, το Plaintext (Αρχικό Κείμενο) και το Ciphertext (Κρυπτοκείμενο) έχουν σχέση ένα προς ένα. Κάθε χαρακτήρας του Plaintext P1 αντιστοιχεί σε έναν και μόνο έναν χαρακτήρα του Ciphertext C1 (χαμηλή Διάχυση). Με τον ίδιο τρόπο γίνεται και η αποκρυπτογράφηση. Αυτοί είναι και οι λόγοι που η ασφάλεια αυτής της προσέγγισης είναι χαμηλή.

Έτσι, η κρυπτογράφηση και η αποκρυπτογράφηση είναι ανεξάρτητες μεταξύ τους, συνεπώς και οι δύο μπορούν να γίνουν εν παραλλήλω. Επίσης, η διάδοση σφάλματος είναι μηδενική, μιας και αν υπάρχει κάποιο λάθος σε ένα block, αυτό δεν επηρεάζει τα υπόλοιπα.



Εικόνα 4.5
Κρυπτογράφηση ECB

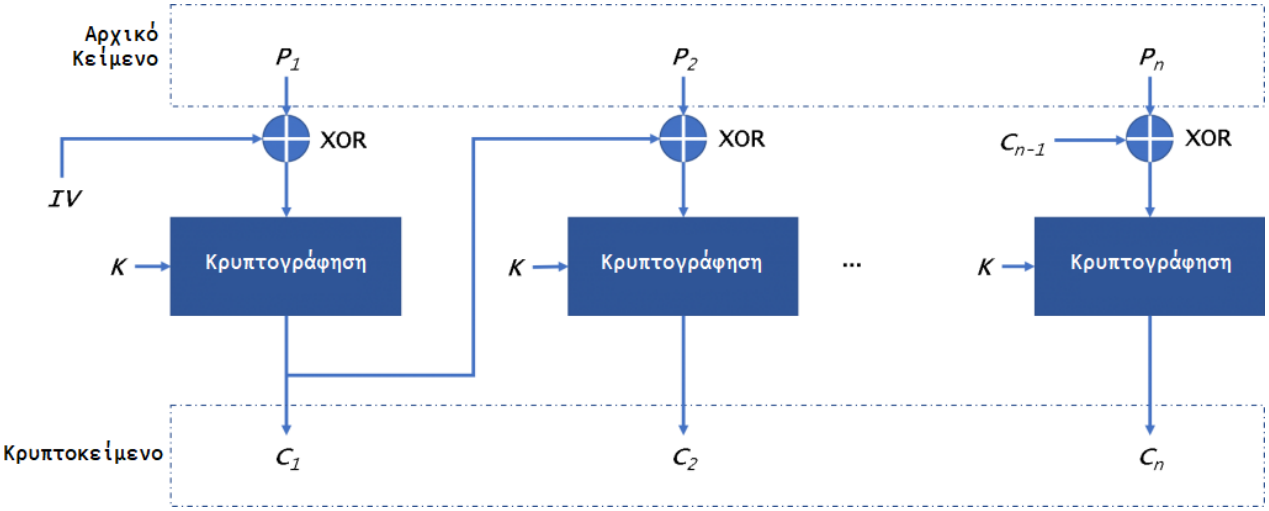


Εικόνα 4.6
Αποκρυπτογράφηση ECB

4.3.2 - CBC: Cipher Block Chaining

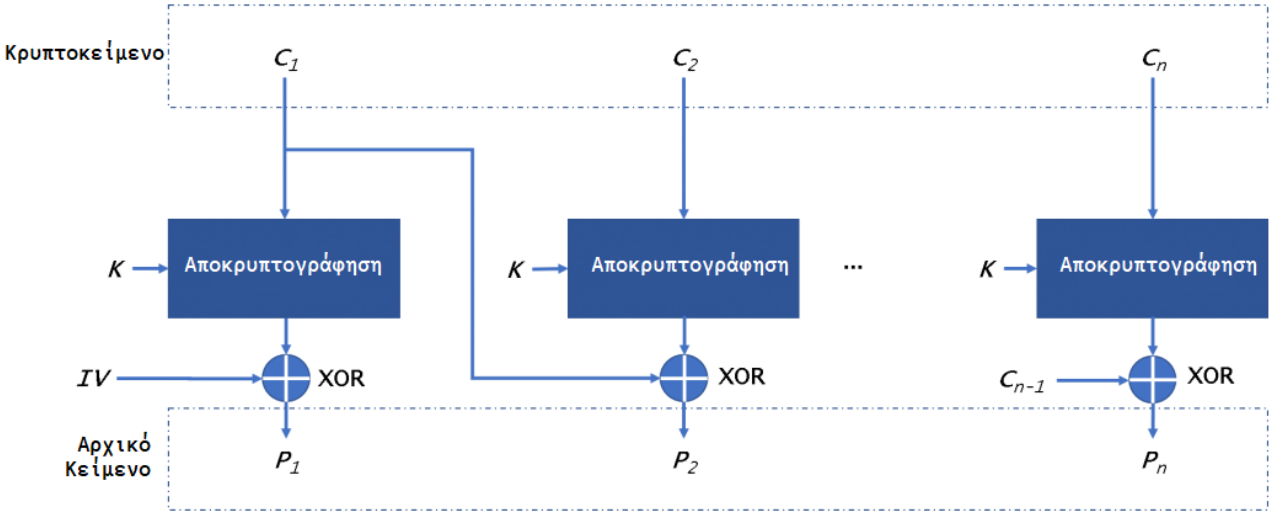
Ο τρόπος CBC για να κρυπτογραφήσει ένα block, πραγματοποιεί αποκλειστική διάζευξη (XOR) του κλειδιού με το προηγούμενο block. Έτσι, αν στην πορεία των γύρων κρυπτογραφηθούν πανομοιότυπα κομμάτια το αποτέλεσμα δε θα είναι το ίδιο κρυπτοκείμενο.

Επειδή όμως το πρώτο block δεν έχει προηγούμενο, χρησιμοποιείται ένας Αρχικός Πίνακας (Initialization Vector). Κι εδώ τα κομμάτια, συμπεριλαμβανομένου του Initialization Vector, συμπληρώνονται μέχρι να αποκτήσουν το μέγεθος του AES. Φαίνεται στη σχηματική αναπαράσταση κάτωθι ότι το πρώτο κομμάτι Plaintext P1 υφίσταται XOR με τον Initialization Vector (IV). Όπως προαναφέρθηκε, έχουν προηγουμένως προστεθεί (ή αναλόγως, αφαιρεθεί) δεδομένα ώστε και τα δύο αυτά να έχουν το μέγεθος AES που επιλέχθηκε (padding). Ύστερα, το προκύπτον κομμάτι κρυπτογραφείται με το κλειδί K. Έτσι προκύπτει το Ciphertext C1. Το C1, εκτός από κομμάτι που θα χρησιμεύσει στο τέλος στην αποκρυπτογράφηση, συνεχίζει να παίρνει μέρος στη διαδικασία της κρυπτογράφησης αφού τα δεδομένα του υφίστανται εκ νέου XOR με το επόμενο κομμάτι Plaintext P2. Ο δεύτερος γύρος κρυπτογράφησης γίνεται με τον ίδιο τρόπο.



Εικόνα 4.7
Κρυπτογράφηση CBC

Η κρυπτογράφηση γίνεται μόνο σειριακά, μιας και για να κρυπτογραφήσουμε ένα κομμάτι πρέπει να αναμένουμε το προηγούμενο, όμως η αποκρυπτογράφηση μπορεί να γίνει και παράλληλα. Επίσης πρέπει να αναφερθεί ότι στον CBC τρόπο το σφάλμα διαδίδεται, δηλαδή ένα σφάλμα που θα δημιουργήσει ένα εσφαλμένο κομμάτι θα επηρεάσει τα επόμενα.

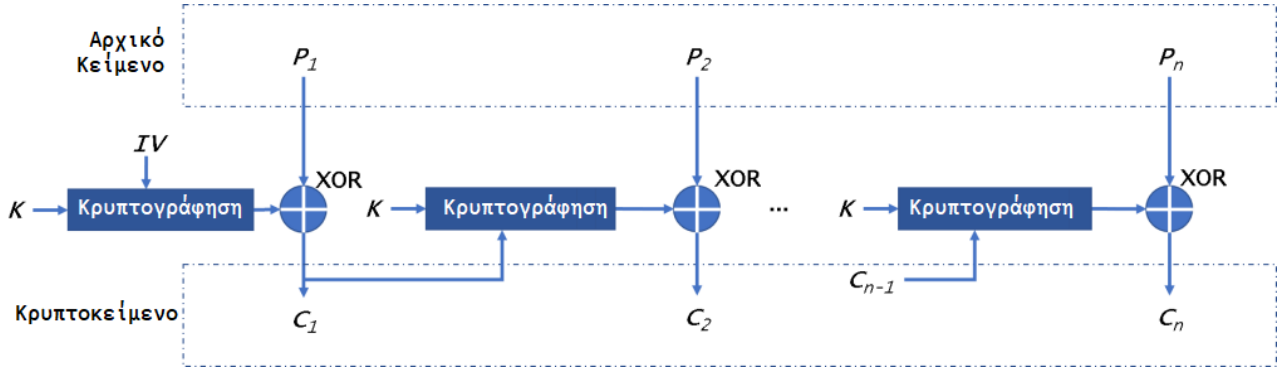


Εικόνα 4.8
Αποκρυπτογράφηση CBC

4.3.3 - CFB: Cipher FeedBack

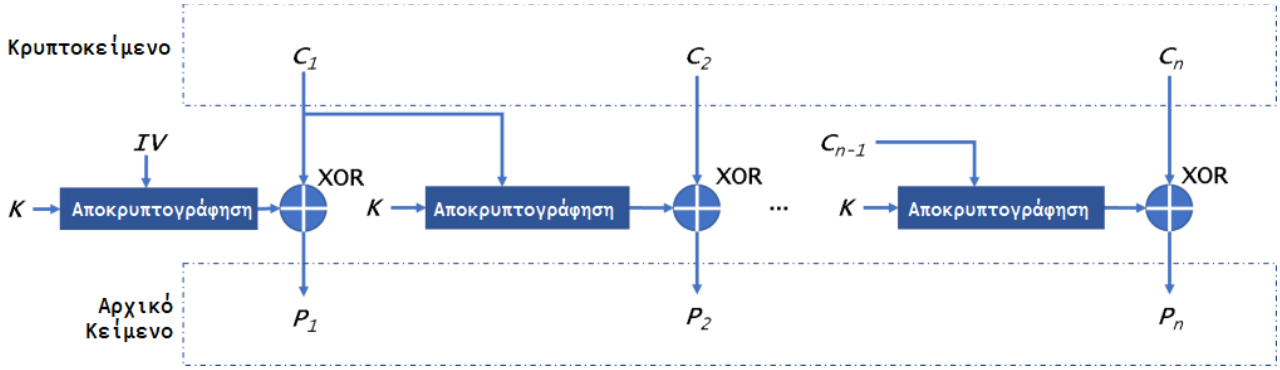
Αυτή είναι η προσέγγιση του AES που χρησιμοποιήθηκε για τις ανάγκες της παρούσας Διπλωματικής Εργασίας. Και στη CFB χρειάζεται Initialization Vector. Η κρυπτογράφηση κομματιών πραγματοποιείται ως Stream Cipher (Κρυπταλγόριθμος Ροής).

Αρχικά κρυπτογραφείται με το κλειδί K ο Initialization Vector και μετά γίνεται XOR με το Plaintext P1 ώστε να παραχθεί το πρώτο κομμάτι του Ciphertext C1. Δε χρειάζεται από την αρχή να συμπληρωθούν δεδομένα ώστε να δημιουργηθούν κομμάτια 128, 192 ή 256 bits γιατί δεν κρυπτογραφείται απευθείας το Plaintext όπως προηγουμένως. Εδώ χρησιμοποιείται Ciphertext (που είναι ήδη στο σωστό μέγεθος) το οποίο υφίσταται XOR με το Plaintext ώστε να παραχθεί το κάθε φορά επόμενο κρυπτογραφημένο κομμάτι. Αυτό, θα κρυπτογραφηθεί με το κλειδί K και μετά γίνεται XOR με το επόμενο κομμάτι του Plaintext P2 ώστε να προκύψει το επόμενο Ciphertext C2.



Εικόνα 4.9
Κρυπτογράφηση CFB

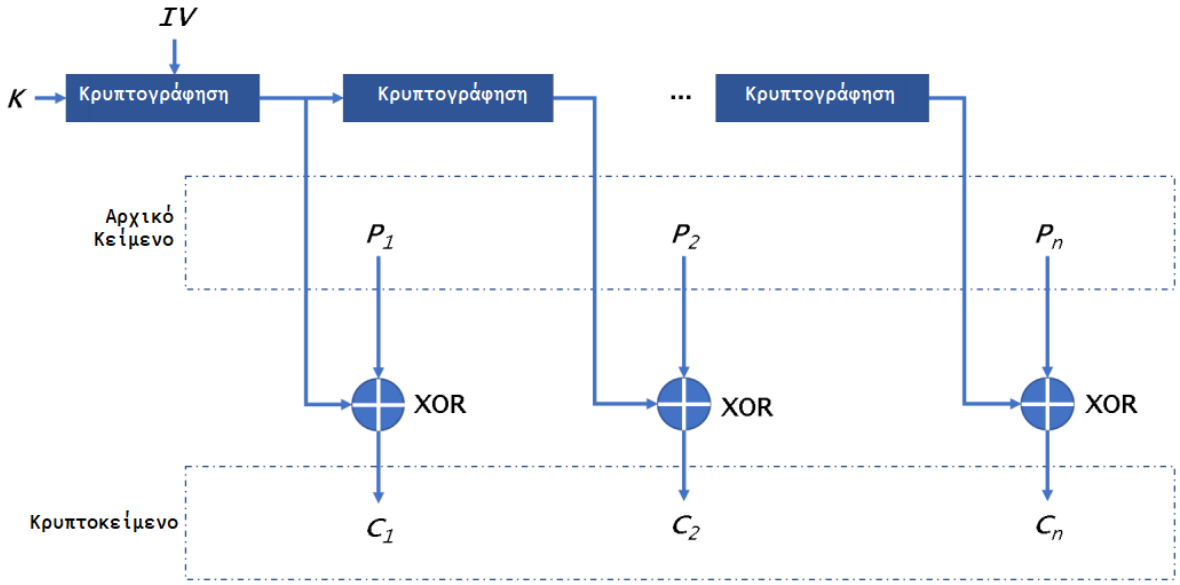
Ομοίως με τον τρόπο λειτουργίας CBC, επειδή ακριβώς αναμένεται κάθε φορά να παραχθεί ένα κομμάτι Ciphertext για να χρησιμοποιηθεί σε κάθε επόμενο βήμα, η διαδικασία δεν παραλληλοποιείται στην κρυπτογράφηση. Στην αποκρυπτογράφηση όμως είναι δυνατή. Ακόμη, πρέπει να αναφερθεί ότι κι εδώ το σφάλμα διαδίδεται, όπως και στη CBC.



Εικόνα 4.10
Αποκρυπτογράφηση CFB

4.3.4 - OFB: Output FeedBack

Στον πρώτο γύρο, κρυπτογραφείται ο Initialization Vector με το κλειδί K. Το προσωρινό block που θα προκύψει υφίσταται XOR με το πρώτο κομμάτι Plaintext P1 κι έτσι προκύπτει το πρώτο κομμάτι Ciphertext C1. Το προσωρινό block που αναφέρθηκε προηγουμένως επανακρυπτογραφείται με το κλειδί K και χρησιμοποιείται σε XOR με το επόμενο κομμάτι Plaintext P2 για να προκύψει το δεύτερο κομμάτι Ciphertext C2. Είναι διαφορετικά από τον τρόπο λειτουργίας CFB διότι κρυπτογραφείται και ο Initialization Vector.

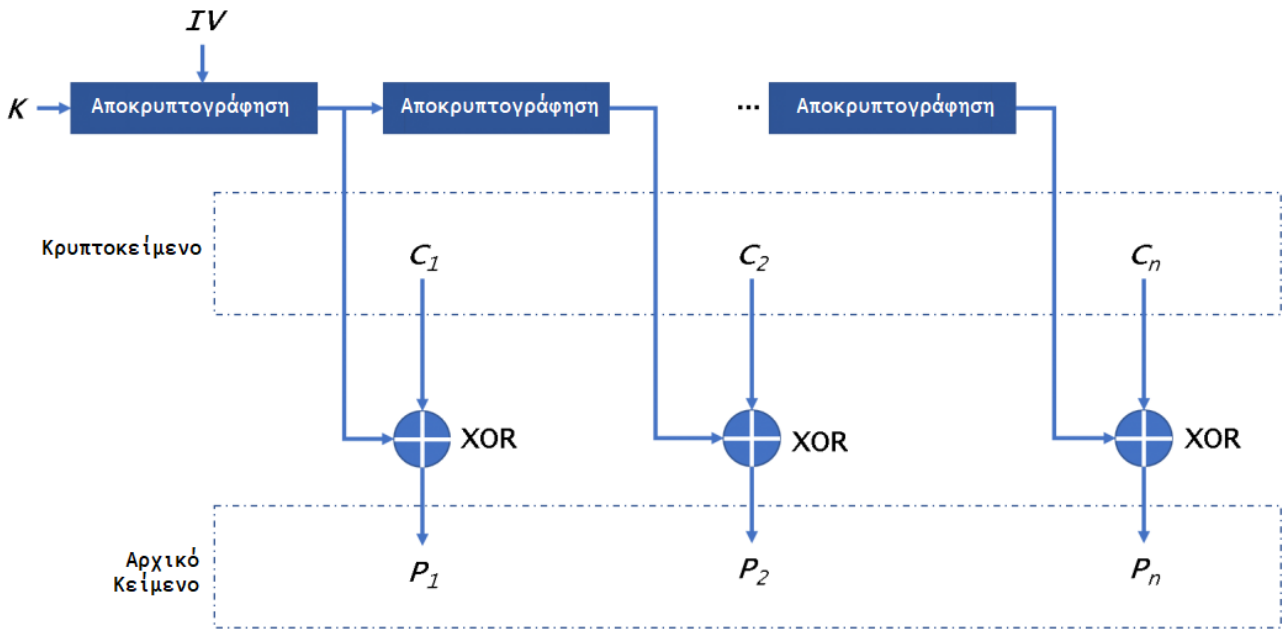


Εικόνα 4.11
Κρυπτογράφηση OFB

Ούτε η κρυπτογράφηση ούτε η αποκρυπτογράφηση του Initialization Vector μπορούν να γίνουν παράλληλα. Άρα δε μπορεί να χρησιμοποιηθεί κάποιο κρυπτογραφημένο προσωρινό block του Initialization Vector σε αποκρυπτογράφηση.

Ένα σημαντικό σημείο της OFB προσέγγισης είναι φανερό στη σχηματική αναπαράσταση ‘ το κάθε κομμάτι Plaintext δε χρησιμοποιείται ξανά πουθενά σε κάποιο επόμενο βήμα άρα υπάρχει μηδενικό σφάλμα διάδοσης. Η λάθος κρυπτογράφηση κάποιου κομματιού δεν επηρεάζει τα επόμενα blocks.

Η κρυπτογράφηση κομματιών χρησιμοποιείται κι εδώ με συνεχή ροή. Επίσης δεν απαιτείται συμπλήρωση κομματιού ώστε να έχει το block συγκεκριμένο μέγεθος.

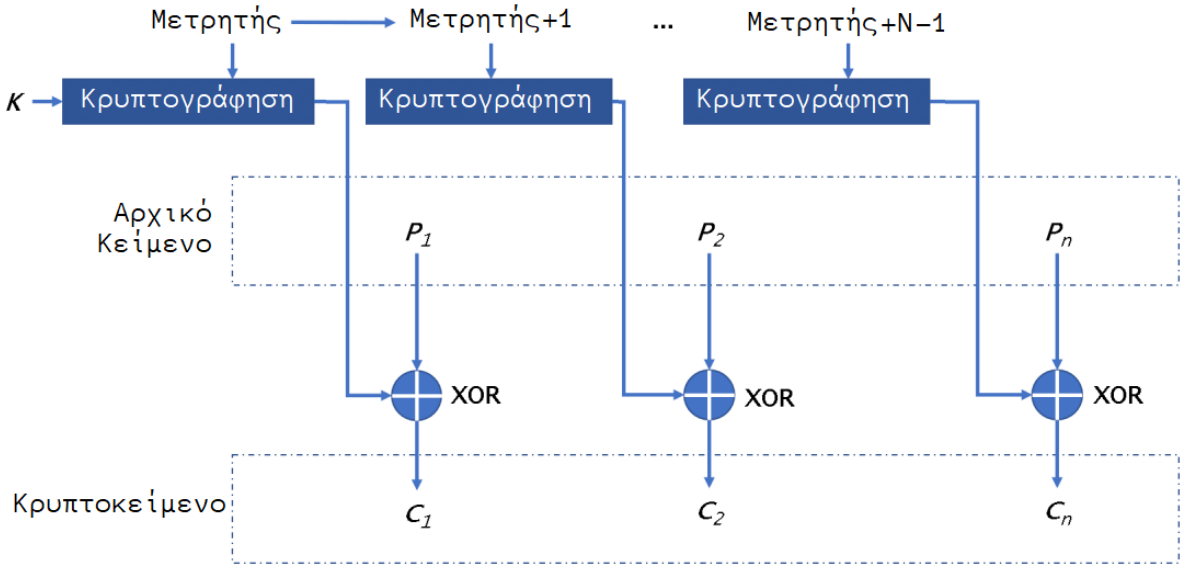


Εικόνα 4.12
Αποκρυπτογράφηση OFB

Θεωρείται ασφαλής για επίθεση επιλεγμένου κειμένου (CPA – Chosen Plaintext Attack) αλλά σχετικά ευάλωτη στις επιθέσεις επιλεγμένου κρυπτοκειμένου (CCA – Chosen Ciphertext Attack) και πρόβλεψης συμπλήρωσης (PA – Padding Oracle Attack).

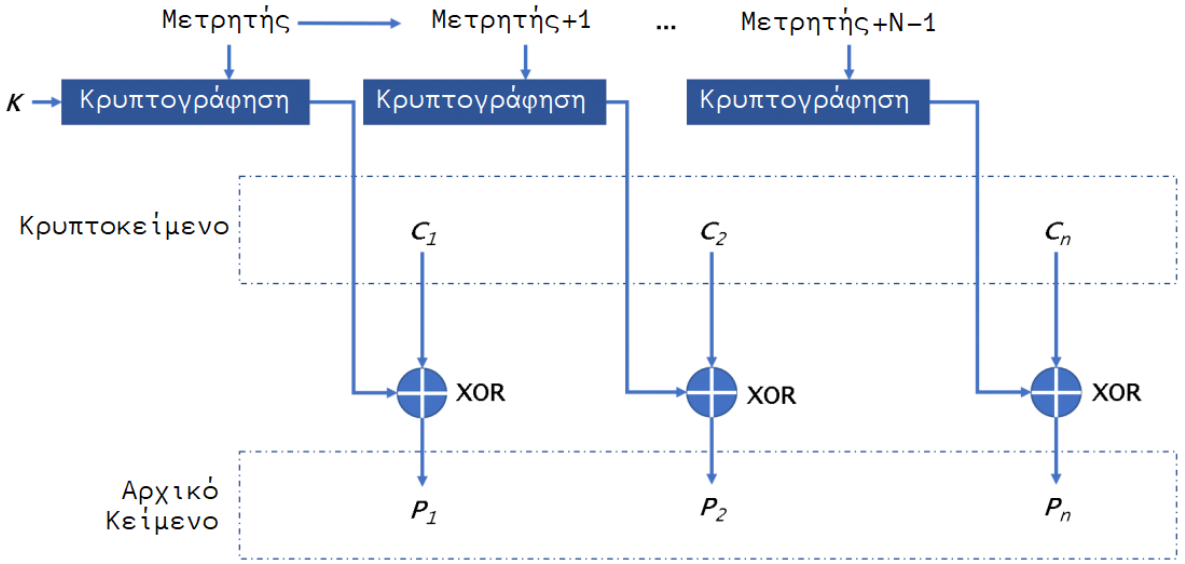
4.3.5 - CTR: Counter

Με αυτόν τον τρόπο λειτουργίας του AES, που επιτρέπει τη συνεχή ροή, χρησιμοποιείται ένας αυξανόμενος Μετρητής και όχι Initialization Vector. Αυτός έχει το ίδιο μέγεθος με το block. Η κρυπτογράφηση του κλειδιού K και του Μετρητή παράγει κάθε φορά ένα προσωρινό block. Η πράξη XOR πραγματοποιείται με το κομμάτι Plaintext και αυτό το προσωρινό block. Ο Μετρητής αυξάνεται και η διαδικασία επαναλαμβάνεται με τον ίδιο τρόπο στον επόμενο γύρο.



Εικόνα 4.13
Κρυπτογράφηση CTR

Όλα τα block χρησιμοποιούν το ίδιο κλειδί. Μοιάζει λίγο με τον OFB αλλά εδώ με το Μετρητή κρυπτογραφείται το κλειδί διαφορετικά σε κάθε γύρο. Σε άλλες προσεγγίσεις ο Αρχικός Πίνακας αλλά και κάθε επόμενος που θα προκύπτει ανά γύρο είναι άγνωστος μέχρι να δημιουργηθεί το προηγούμενο block που τον παράγει. Εδώ στον CTR, αν υπάρχει πρόσβαση στην τιμή του Μετρητή (προφανώς και του κλειδιού K), είναι δυνατή και η κρυπτογράφηση αλλά και η αποκρυπτογράφηση δεδομένων εν παραλλήλω (Wang, 2019).



Εικόνα 4.14
Αποκρυπτογράφηση CTR

4.4 - Η ΠΡΟΣΕΓΓΙΣΗ ΤΗΣ ΠΑΡΟΥΣΑΣ ΔΙΠΛΩΜΑΤΙΚΗΣ ΕΡΓΑΣΙΑΣ - CFB

Υπάρχουν οι προσεγγίσεις των 128, 192 και 256 bits. Τα bit αυτά αναφέρονται στο μέγεθος *κλειδιού*. Είναι προφανές ότι όσο περισσότερα είναι τα bit κλειδιού, τόσα περισσότερα διαφορετικά κλειδιά μπορούμε να ορίσουμε. Στην παρούσα διπλωματική εργασία θα χρησιμοποιηθούν 256 bit για το κλειδί.

Ο αλγόριθμος χρησιμοποιεί κουτιά αντικατάστασης ή S-boxes (Substitution boxes) και, ανάλογα με το μέγεθος των τμημάτων και το μέγεθος του αρχικού κλειδιού, πραγματοποιεί 10, 12 ή 14 επαναλήψεις.

Ο ψευδοκώδικας κάθε επανάληψης του αλγόριθμου είναι ως εξής:

```
Round(state, keyOfRound) {
        byteSubstitution(state);
        shiftRow(state);
        mixColumn(state);
        addRoundKey(state, keyOfRound);
}
```

Η συνάρτηση αυτή εκτελείται σε κάθε γύρο που εκτελεί ο αλγόριθμος. Οι επιμέρους συναρτήσεις έχουν τις εξής λειτουργίες:

– `byteSubstitution(state)`

Αποτρέπει τη γραμμική και τη διαφορική κρυπτανάλυση. Είναι μη γραμμικό μέρος (είναι δυνατόν να παραλληλοποιηθεί).

– `shiftRow(state)`

Αυξάνει, μετακινώντας γραμμές των s-boxes, τη σύγχυση του κρυπτοκειμένου. Είναι γραμμικό μέρος (δεν παραλληλοποιείται).

– `mixColumn(state)`

Αυξάνει, αλλάζοντας τη θέση των στηλών των s-boxes, τη σύγχυση του κρυπτοκειμένου. Είναι γραμμικό μέρος. Στην τελευταία επανάληψη της Round(), δηλαδή στον τελευταίο γύρο επανάληψης δεν γίνεται κλήση της mixColumn() διότι χρησιμοποιείται ως κείμενο εισόδου το αρχικό κείμενο, ενώ η έξοδος είναι το κρυπτοκείμενο.

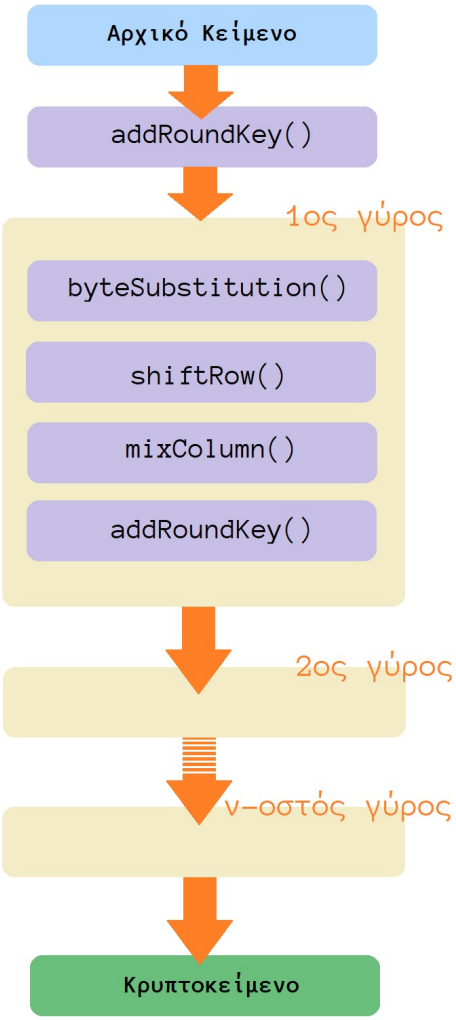
– `addRoundKey(state, keyOfround)`

Προσθέτει, με XOR το κλειδί του τρέχοντος γύρου στην έξοδο της mixColumn().

Ο ψευδοκώδικας του πλήρους αλγόριθμου είναι ως εξής:

```
AES(state, key){
        keyPadding(key);
        addRoundKey(state, key);
        for(i=1; i<numberOfRounds; i++){
                Round(state, key + numberOfBox * i);
        }
        byteSubstitution(state);
        shiftRow(state);
        addRoundKey(state, key + numberOfBox * i);
}
```

Η επιμέρους συνάρτηση `keyPadding(key)` προσαρμόζει το μέγεθος του κλειδιού, ανάλογα με την προσέγγιση που επιλέχθηκε. Για παράδειγμα, αν οριστεί κλειδί των 128 bits (δηλαδή 16 bytes), και εισαχθεί τελικώς κλειδί με 10 bytes, θα προστεθεί στο κλειδί αυτό μια κατάληξη (επιλεγμένη προγραμματιστικά , πχ "{") ώστε να συμπληρωθούν τα υπόλοιπα 6. Μετά, αυτό το συμπληρωμένο με bytes κλειδί εισάγεται στην αλγόριθμο. Αν πάλι το κλειδί είναι μεγαλύτερο από 16 bytes, αποκόπτονται δεδομένα από το τέλος (Κάτος, Στεφανίδης, 2003).



Εικόνα 4.15
Σχηματική αναπαράσταση λειτουργιών των συναρτήσεων του AES

ΚΕΦΑΛΑΙΟ 5
ΛΕΙΤΟΥΡΓΙΚΟ ΣΥΣΤΗΜΑ GNU/Linux

5.1 – ΑΠΑΡΧΕΣ GNU ΚΑΙ LINUX

Το Raspberry Pi κατά το χρόνο συγγραφής της παρούσας διπλωματικής εργασίας κατασκευάστηκε για να τρέχει το λειτουργικό σύστημα Raspberry Pi OS, που βασίζεται σε κάποια έκδοση Linux. Όμως, η ολοκληρωμένη ονομασία του είναι GNU/Linux, καθώς απαρτίζεται από δύο στοιχεία. Τις βιβλιοθήκες/εφαρμογές του και τον πυρήνα του, τα οποία δημιουργήθηκαν από δύο ανεξάρτητες οντότητες.

Το Free Software Movement (Κίνημα Ελεύθερου Λογισμικού) ανακοινώθηκε επίσημα το 1983 από τον Richard Stallman ο οποίος εξέδωσε το σχετικό γραπτό μανιφέστο ένα χρόνο αργότερα. Σκοπός του κινήματος ήταν η περιφρούρηση των ελευθεριών των χρηστών λογισμικού. Για να θεωρείται το λογισμικό ελεύθερο πρέπει να διακατέχεται από τέσσερις ελευθερίες.

✔ Η εκτέλεση του λογισμικού όπως ο χρήστης επιθυμεί, για οποιοδήποτε σκοπό.
✔ Η μελέτη του τρόπου λειτουργίας του λογισμικού και η τροποποίησή του για τις ανάγκες του προγραμματιστή. Απαραίτητη προϋπόθεση είναι η πρόσβαση στον πηγαίο κώδικά του.
✔ Η αναδιανομή αντιγράφων προς βοήθεια άλλων.
✔ Η αναδιανομή αντιγράφων των επεξεργασμένων εκδόσεων του λογισμικού σε άλλους. Απαραίτητη προϋπόθεση είναι η πρόσβαση στον επεξεργασμένο πηγαίο κώδικα.

Εικόνα 5.1
Ο Richard Stallman το 1983
https://www.timetoast.com/timelines/the-life-of-richard-stallman

Το 1984 ο ίδιος ηγήθηκε μιας ομάδας προγραμματιστών που σκοπός τους ήταν η δημιουργία ελεύθερου λογισμικού, εναλλακτικού από αυτό που περιεχόταν στο Unix, κι έτσι ξεκίνησε το GNU Project. Οι ιθύνοντες του GNU Project προσπαθούσαν να δημιουργήσουν ένα ελεύθερο λειτουργικό σύστημα ξεκινώντας από τις βιβλιοθήκες, τα βοηθητικά προγράμματα και τις εφαρμογές (Wikipedia, 2022), (www.gnu.org).

Το 1989 δημοσιεύει την GNU General Public Licence, μια τεκμηριωμένη σειρά αδειών χρήσης που αφορά το ελεύθερο λογισμικό, περιφρουρώντας τις τέσσερις παραπάνω ελευθερίες του Free Software Foundation. Μέχρι το 1990 είχαν δημιουργηθεί πάρα πολλά στοιχεία για τη σύνθεση ενός ολοκληρωμένου λειτουργικού συστήματος, αλλά τους έλειπε ο πυρήνας.

Εικόνα 5.2
Ο Linus Torvalds
https://openslime.it/2018/10/linus-torvalds-linux-ritiro/

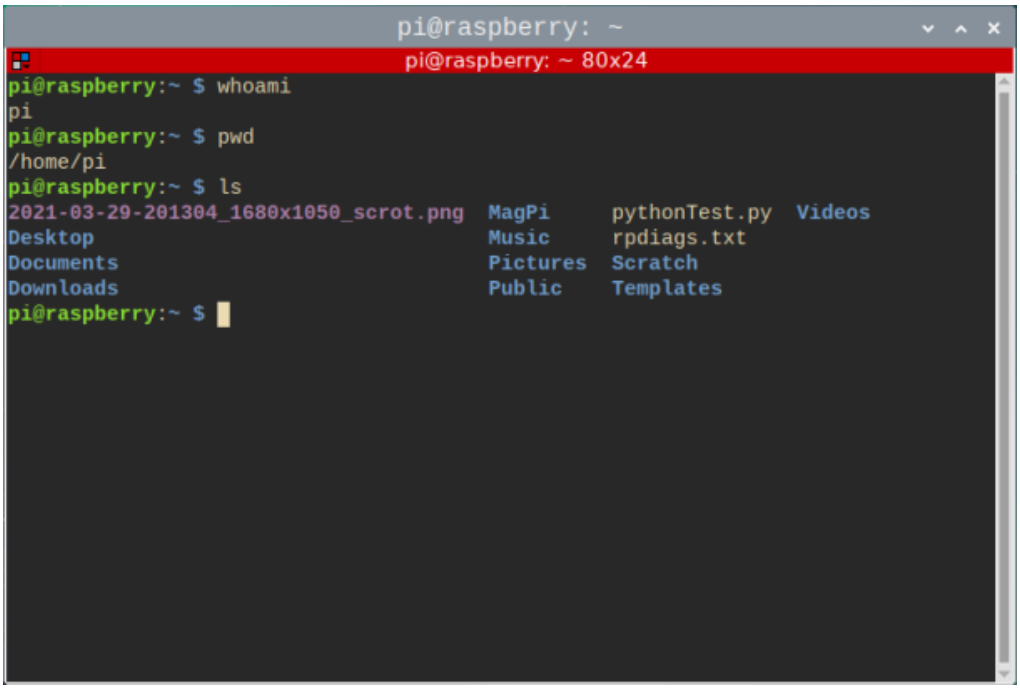
Το 1991, οκτώ χρόνια μετά την εκκίνηση του GNU Project, ένας άλλος προγραμματιστής ονόματι Linus Torvalds που εκείνη την περίοδο φοιτούσε στο Πανεπιστήμιο του Ελσίνκι, εργαζόταν πάνω σε αυτό ακριβώς το αντικείμενο. Ο Torvalds δημιούργησε αυτόν τον πυρήνα – υπόστρωμα και σύνδεσε τα μέχρι στιγμής κομμάτια του GNU Project σε ένα ολοκληρωμένο λειτουργικό σύστημα. Ανέβασε τον κώδικα (που περιλάμβανε μόλις 9700 γραμμές) στο server του Πανεπιστημίου υπό το όνομα “Freax”. Όμως, ο διαχειριστής του ftp server του Πανεπιστημίου το βρήκε πολύ αστείο και άλλαξε στον server το όνομα του φακέλου σε Linux, μια παράφραση του ονόματος του Torvalds και των Unix.

Το λειτουργικό σύστημα που προέκυψε ονομάστηκε GNU/Linux, καθώς είναι συνδυασμός των δύο παραπάνω παραγόντων, του πυρήνα του Torvalds και των στοιχείων του GNU.

5.2 - ΚΕΛΥΦΗ

Ο πυρήνας (kernel) είναι το στοιχείο που εκτελεί λειτουργίες πιο κοντά στο hardware και είναι κατά κύριο λόγο κλειστός στις εξωτερικές παρεμβάσεις. Όμως, μιας και πρόκειται για open source software (λογισμικό ανοιχτού κώδικα), είναι εν τέλει δυνατή η τροποποίησή του αλλά αντενδείκνυται για μη πεπειραμένους χρήστες και απαιτεί αυξημένα δικαιώματα. Για την επικοινωνία με τον πυρήνα χρησιμοποιούνται τα κελύφη.

Το κέλυφος είναι ένα interface (διεπαφή) του πυρήνα του λειτουργικού συστήματος με το χρήστη. Όποτε ο χρήστης ανοίγει ένα τερματικό, ο πυρήνας τρέχει μια νέα οντότητα κελύφους. Διαμέσω αυτού μπορούν να εκτελεστούν προγράμματα και εφαρμογές στον πυρήνα. Έτσι, είναι δυνατόν εκτός από επιπρόσθετο λογισμικό που τυχόν έχει εγκατασταθεί, να εκμεταλλευθούν και οι δυνατότητες του ίδιου του λειτουργικού συστήματος.



Εικόνα 5.3

Το κέλυφος BASH μέσα από τον προσομοιωτή τερματικού Terminator

Υπάρχουν πέντε βασικά είδη κελύφους:

5.2.1 – Bourne Shell (sh)

Δημιουργήθηκε στα AT&T Labs από τον Steve Bourne και θεωρείται το πρώτος κέλυφος σε περιβάλλον UNIX. Η δημοτικότητά του οφειλόταν στο μικρό μέγεθος απαιτήσεων και στην ταχύτητά του. Όμως, το Bourne Shell έχει ορισμένα μειονεκτήματα.

- Δεν έχει εγγενείς λειτουργίες για το χειρισμό λογικών αριθμητικών πράξεων.
- Δε μπορεί να ανακτήσει προηγούμενες εκτελεσθείσες εντολές.
- Του λείπουν χαρακτηριστικά κατανοησιμότητας που επιτρέπουν μια εύκολη διαδραστική χρήση.
- Το absolute path για το κέλυφος αυτό είναι /bin/sh και /sbin/sh. Από προεπιλογή, χρησιμοποιεί την προτροπή $ για τον απλό χρήστη και # για το διαχειριστή.

5.2.2 – C Shell (csh)

Δημιουργήθηκε στο Πανεπιστήμιο της Καλιφόρνια από τον Bill Joy. Περιλαμβάνει εντολές για το χειρισμό αριθμητικών πράξεων και έχει σύνταξη παρόμοια με της γλώσσας C. Κρατάει επίσης ιστορικό των εντολών. Άλλη μια χρήσιμη λειτουργία του είναι τα ψευδώνυμα (aliases) που επιτρέπουν την προσωρινή ή μόνιμη αποθήκευση σύνθετων εντολών με μικρά ονόματα. Το absolute path για το κέλυφος αυτό είναι /bin/csh. Από προεπιλογή, χρησιμοποιεί την προτροπή hostname% για τον απλό χρήστη και hostname# για το διαχειριστή.

5.2.3 – Korn shell (ksh)

Υλοποιήθηκε στα AT&T Labs από τον David Korn για να βελιτώσει το Bourne shell, ως άμεσο υπερσύνολό του. Επιτρέπει στους χρήστες τη χρήση νέων εντολών όπως υποστήριξη αριθμητικών πράξεων, ενώ προσφέρει διαδραστικά χαρακτηριστικά, παρόμοια με το C shell κι επίσης το χειρισμό συμβολοσειρών, πινάκων και συναρτήσεων, όπως στη γλώσσα προγραμματισμού C. Επιπροσθέτως, είναι πιο γρήγορο από άλλα κελύφη, συμπεριλαμβανομένου του C shell. Το absolute path για το κέλυφος αυτό είναι /bin/ksh. Από προεπιλογή, χρησιμοποιεί την προτροπή $ για τον απλό χρήστη και # για το διαχειριστή.

5.2.4 - Bourne-Again Shell (bash)

Επίσης γνωστό ως bash shell, σχεδιάστηκε για να είναι συμβατό με το bourne shell. Επισης, περιλαμβάνει χρήσιμα χαρακτηριστικά από διάφορα άλλα κελύφη, όπως το Korn shell και το C shell. Οι προηγούμενες εκτελεσθείσες εντολές αποθηκεύονται σε ιστορικό και μπορεί να γίνει χρήση ξανά χωρίς επαναπληκτρολόγηση. Το absolute path για το κέλυφος αυτό είναι /bin/bash. Από προεπιλογή, χρησιμοποιεί την προτροπή bash-versionNumber$ για τον απλό χρήστη και bash-versionNumber# για το διαχειριστή.

5.2.5 - Z Shell (zsh)

Εδώ υπάρχουν πολλές βελτιώσεις από τα προηγούμενα είδη, με συμπεριλαμβανόμενες τις λειτουργίες από τα άλλα κελύφη. Για παράδειγμα, δημιουργία αρχείων υπό συγκεκριμένες προϋποθέσεις, υποστήριξη θεμάτων και plugin, ευρετήριο native λειτουργιών, αυτόματη συμπλήρωση εντολών, και άλλα (Digital Ocean, 2022).

ΚΕΦΑΛΑΙΟ 6
PYTHON

6.1 – ΣΧΕΤΙΚΑ ΜΕ ΤΗΝ PYTHON

Η Python είναι μια αντικειμενοστρεφής γλώσσα προγραμματισμού υψηλού επιπέδου, δηλαδή η σύνταξή της θυμίζει την ανθρώπινη, φυσική γλώσσα.

Έχει κομψή εμφάνιση που καθιστά τα προγράμματα ευανάγνωστα. Είναι απλή στη σύνταξη κι έτσι η συγγραφή και η αποσφαλμάτωση καθίστανται εύκολες και διαχειρίσιμες. Μπορεί να εμπλουτιστεί με νέα πακέτα (packages) ή μονάδες (modules) από άλλες γλώσσες προγραμματισμού όπως η C και η C++.

Διαθέτει μεγάλη ποικιλία από προεγκατεστημένες βιβλιοθήκες που καλύπτουν ένα μεγάλο εύρος από κοινά προγραμματιστικά θέματα όπως εγγραφή και ανάγνωση αρχείων, αναζήτηση κειμένου με τη χρήση regular expressions και σύνδεση με servers ιστοσελίδων.

Η διαδραστική λειτουργία που διαθέτει επιτρέπει την εύκολη δοκιμή κομματιών κώδικα. Μπορεί να ενσωματωθεί σε εφαρμογές για να παρέχει προγραμματισμένο interface. Τρέχει στα δημοφιλέστερα λειτουργικά συστήματα, συμπεριλαμβανομένων των περιβάλλοντων Windows, Linux, Unix, Mac OS X, με ανεπίσημη μεταφορά σε Android και iOS.

Είναι δωρεάν, αφενός μεν επειδή μπορεί οποιοσδήποτε να την κατεβάσει, να τη χρησιμοποιήσει και να τη συμπεριλάβει στην εφαρμογή του, αφετέρου δε λόγω του γεγονότος ότι μπορεί ελεύθερα να τροποποιηθεί και να διοχετευθεί διότι είναι κατοχυρωμένη με άδεια λογισμικού ανοιχτού κώδικα.

Εικόνα 6.1
Λογότυπο της Python
https://www.python.org/static/community_logos/python-logo-master-v3-TM.png

Η Python καθαυτή έχει ορισμένα πολύ ενδιαφέροντα χαρακτηριστικά. Όπως άλλες διάσημες γλώσσες, διαθέτει αρχέτυπες μορφές και δομές δεδομένων όπως αριθμοί (κινητής υποδιαστολής, σύνθετοι και ακέραιοι μεγάλου μήκους), συμβολοσειρές, λίστες και λεξικά. Ακόμη, ως αντικειμενοστρεφής, υποστηρίζει κλάσεις, πολλαπλή κληρονομικότητα και μιας μικρής κλίμακας ενθυλάκωση.

Ο κώδικας μπορεί να ομαδοποιηθεί προγραμματιστικά σε μονάδες (modules) ή πακέτα (packages). Υποστηρίζεται η έγερση και σύλληψη εξαιρέσεων που επιτρέπει τον καθαρότερο χειρισμό σφαλμάτων.

Οι τύποι δεδομένων ορίζονται ρητά και δυναμικά. Ανάμειξη τύπων δεδομένων (όπως η πρόσθεση αριθμού με συμβολοσειρά) δημιουργεί exception, κι έτσι τα λάθη ανακαλύπτονται γρηγορότερα.

Ακόμη, η Python προσφέρει εξεζητημένα προγραμματιστικά εργαλεία όπως οι Generators (Γεννήτριες) και η List Comprehension (Κατανόηση Λίστας). Με έναν Generator υλοποιείται μια τεχνική που αναφέρεται και ως "χαλαρή μέτρηση", δημιουργείται δηλαδή ένας προσωρινός μετρητής που δεν καλείται παρά μόνο όταν χρειάζεται και ύστερα απελευθερώνεται, εξοικονομώντας μνήμη και αποφεύγοντας διπλές μετρήσεις. Το List Comprehension χρησιμοποιεί συμπτυγμένη σύνταξη ώστε να δημιουργεί μια νέα λίστα

βασισμένη στις τιμές μιας υπάρχουσας λίστας.

Η Python έχει ενεργή μια μεγάλη παγκόσμια κοινότητα υποστήριξης, ενώ έχουν εκδοθεί διάφοροι συντακτικοί οδηγοί που ορίζουν άτυπα ένα συγκεκριμένο τρόπο συγγραφής της Python για την καλύτερη συνεννόηση και τήρηση καλών πρακτικών, όπως ο PEP-8.

Η Python χρησιμοποιεί διερμηνευτή και όχι μεταγλωττιστή. Αυτό σημαίνει ότι δεν απαιτείται το σύνολο κάποιου προγράμματος να έχει περάσει από το στάδιο μεταγλώττισης ' οι εντολές διερμηνεύονται και εκτελούνται απευθείας, προσφέροντας μεγαλύτερη ταχύτητα. Διαθέτει αυτόματο garbage collector (συλλέκτη απορριμάτων), ο οποίος μαρκάρει δεδομένα προς τα οποία δεν υπάρχει ενεργή αναφορά από πουθενά στο πρόγραμμα, δεδομένα τα οποία απελευθερώνει χωρίς την παρέμβαση χρήστη ύστερα από ορισμένες συνθήκες. Η αυτόματη διαχείριση μνήμης απελευθερώνει από την ανάγκη της χειροκίνητης δέσμευσης και απελευθέρωσης μνήμης στον κώδικα. Ο garbage collector καλείται είτε με ρητή εντολή από τον προγραμματιστή είτε με την ανάγκη δέσμευσης μνήμης είτε αυτόματα, αν κάποια δομή δεδομένων θεωρείται ότι δε θα χρησιμοποιηθεί ξανά από το διερμηνευτή.

Εικόνα 6.2
Ο Guido van Rossum το 2018
https://python.developpez.com/actu/283188/Guido-Van-Rossum-le-createur-du-langage-de-programmation-Python-prend-sa-retraite-ca-a-ete-une-aventure-incroyable-dit-il/

Η Python άρχισε να σχεδιάζεται από τον Ολλανδό Guido van Rossum στα τέλη του 1980, ο οποίος εργαζόταν στο Centrum Wiskunde & Informatica (CWI), ως διάδοχος της γλώσσας ABC. Η πρώτη της έκδοση, η 0.9.0, βγήκε μια δεκαετία αργότερα, το 1991. Η έκδοση 2.0 ήταν διαθέσιμη το 2000, της οποίας η τελευταία έκδοση (2.7.18) σταμάτησε την υποστήριξή της το 2020. Αυτή τη στιγμή υποστηρίζεται κάποια αναθεώρηση της έκδοσης 3.x που θα σταματήσει την υποστήριξή της προς το τέλος της δεκαετίας.

Ο Guido von Rossum αποσύρθηκε στις 12 Ιουλίου 2018 από τις υποχρεώσεις του στην Python ως "ισόβιος καλοκάγαθος δικτάτορας", ένας τίτλος που του αποδόθηκε από την κοινότητα για να τιμήσει την πολυετή του δέσμευση στην βασική λήψη αποφάσεων του όλου εγχειρήματος. Τον Ιανουάριο του 2019 συστήθηκε ένα πενταμελές συμβούλιο που έκτοτε έχει

αναλάβει τη συνέχεια.

Αντί για την ενσωμάτωση όλης της λειτουργικότητας στον πυρήνα της, η Python σχεδιάστηκε να έχει μεγάλη επεκτασιμότητα με την ανάπτυξη μεγάλου αριθμού ανεξάρτητων πακέτων. Το γεγονός αυτό κατέστησε δυνατή την πρόσθεση προγραμματιστικών διεπαφών σε ήδη υπάρχουσες εφαρμογές, αυξάνοντας περισσότερο την ευελιξία και δημοφιλία της. Μια χαρακτηριστική αντίθεση υπάρχει μεταξύ του βασικού ρητού της γλώσσας Perl "υπάρχουν πάνω από ένας τρόποι για να το κάνεις" με της Python "θα πρέπει να υπάρχει ένας (κατά προτίμηση μόνο ένας) τρόπος για να το κάνεις".

Στη λίστα TIOBE (από τα αρχικά της κωμωδίας του Oscar Wilde "The Importance of Being Earnest") με τις δημοφιλέστερες γλώσσες προγραμματισμού, είναι στην πρώτη θέση με στοιχεία της 22 Φεβρουαρίου 2024. Η λίστα αντλεί τα στοιχεία της από δημοφιλείς μηχανές αναζήτησης με βάσει τα ερωτήματα που έχουν λάβει.

Feb 2024	Feb 2023	Change	Programming Language	Ratings	Change
1	1		Python	15.16%	-0.32%
2	2		C	10.97%	-4.41%
3	3		C++	10.53%	-3.40%
4	4		Java	8.88%	-4.33%
5	5		C#	7.53%	+1.15%
6	7	↑	JavaScript	3.17%	+0.64%
7	8	↑	SQL	1.82%	-0.30%
8	11	↑	Go	1.73%	+0.61%
9	6	↓	Visual Basic	1.52%	-2.62%
10	10		PHP	1.51%	+0.21%

Εικόνα 6.3
Δημοτικότητα γλωσσών προγραμματισμού (22 Φεβρουαρίου 2024)
https://www.tiobe.com/tiobe-index/

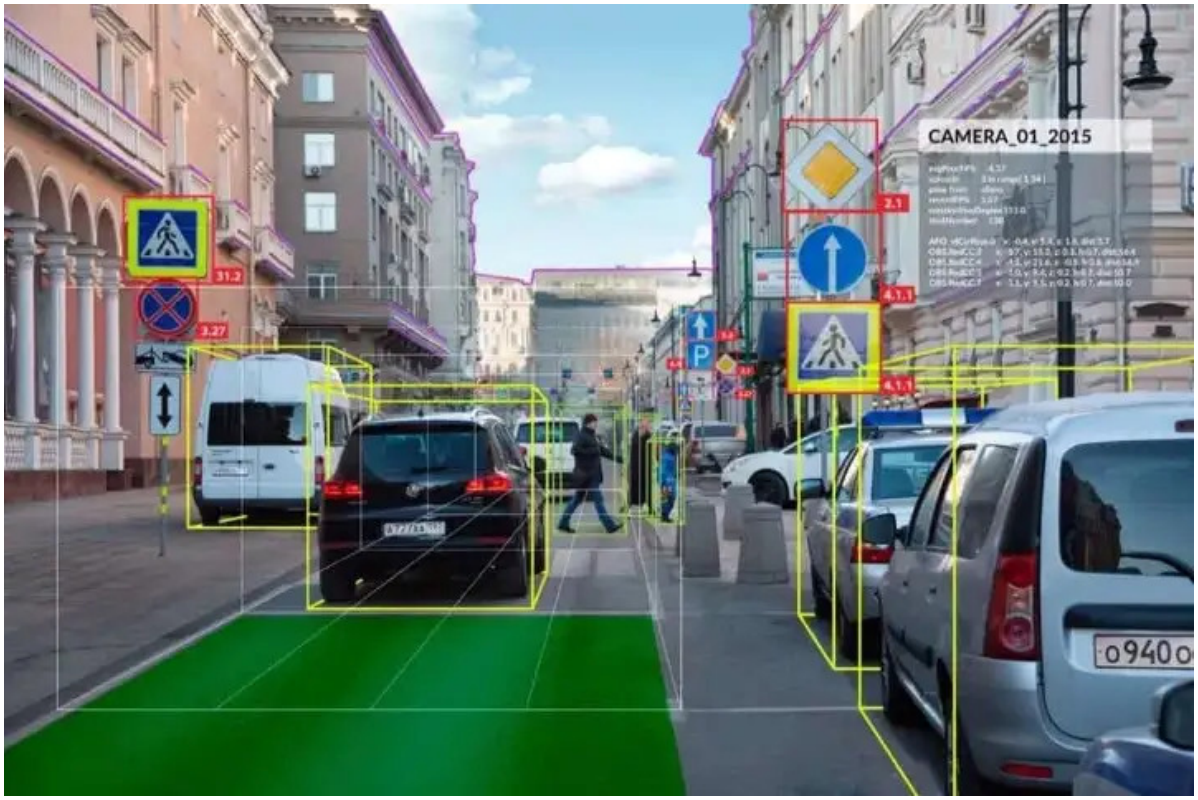
6.2 – OPEN CV (Open Computer Vision Library)

Το OpenCV είναι ένα εργαλείο λογισμικού που επιτρέπει την επεξεργασία βίντεο και εικόνας σε πραγματικό χρόνο, με ικανότητες μηχανικής μάθησης, προσφέροντας παράλληλα αναλυτικές μετρικές. Ουσιαστικά είναι μια βιβλιοθήκη με άδεια χρήσης που παρέχεται από το BSD[2] η οποία παρέχει αλγόριθμους επεξεργασίας 2D και 3D γραφικών οι οποίοι σε διαφορετική περίπτωση είναι προσβάσιμοι μόνο από υψηλών προδιαγραφών επαγγελματικά προϊόντα

2 Το BSD, από το Berkeley Software Distribution, είναι ένα λειτουργικό σύστημα τύπου Unix που δημιουργήθηκε το 1977 από το CSRG (Computer System Research Group), μια ερευνητική ομάδα στο Πανεπιστήμιο Berkeley της California.

λογισμικού.

Με το απλό programming interface του OpenCV είναι δυνατή η εύκολη συγγραφή κομματιών κώδικα που αναλαμβάνουν αφαίρεση παρασκηνίου, φίλτρα, ταυτοποίηση προτύπων και κατηγοριοποίηση εικόνων. Η ανάλυση βίντεο σε πραγματικό χρόνο περιλαμβάνει αναγνώριση και ακολούθηση ζώων (ακόμη και το είδος), αντικειμένων, ατόμων, καθώς και πινακίδων οχημάτων, κι επίσης χαρακτηριστικά προσώπου (Intel, 2017).



Εικόνα 6.4

Εντοπισμός αντικειμένων εικόνας με OpenCV

https://medium.com/analytics-vidhya/object-detection-with-opencv-step-by-step-6c49a9cc1ff0

Είναι γραμμένο σε βελτιστοποιημένη C/C++, διαλειτουργικό από σχεδιασμού του ώστε να τρέχει απροβλημάτιστα σε μια πληθώρα πλατφορμών υλικού. Το OpenCV έχει υλοποιηθεί για χρήση σε C, C++, Python και Java, σε λειτουργικό σύστημα Windows, αρκετές διανομές Linux, macOS, iOS και Android. Αν και οι κάμερες γενικά λειτουργούν καλύτερα με τα αντίστοιχα προγράμματα οδήγησης, ακόμα και η απλή αναγνώρισή τους από το λειτουργικό σύστημα με τους προεπιλεγμένους drivers αρκεί για τη χρησιμοποίηση OpenCV με αυτές.

Αξίζει να σημειωθεί ότι το OpenCV εκμετελλεύεται πλήρως την αρχιτεκτονική πολλών πυρήνων και το OpenCL[3], καθιστώντας έτσι δυνατή τη χρήση hardware acceleration, όταν υπάρχει επεξεργαστής για ενσωματωμένα γραφικά. Χρησιμοποιεί τη βιβλιοθήκη FFMPEG[4].

6.3 – PYTHON SOCKETS

Στην παρούσα διπλωματική εργασία θα τρέξουν δύο ξεχωριστά προγράμματα σε δύο διαφορετικούς υπολογιστές. Επειδή οι ενέργειες αυτές θα γίνονται ταυτόχρονα, είναι αναγκαίος

3 Το OpenCl (Open Computing Language) είναι ένα framework για συγγραφή προγραμμάτων που εκτελούνται σε ετερογενείς πλατφόρμες που αποτελούνται από πολυπύρηνους επεξεργαστές γενικής χρήσης, γραφικών, ψηφιακού σήματος και άλλους επιταχυντές.

4 Λογισμικό ανοιχτού κώδικα που αποτελείται από μια σουίτα βιβλιοθηκών και προγραμμάτων για το χειρισμό βίντεο, ήχου και άλλων αρχείων πολυμέσων.

ο συντονισμός των εργασιών μεταξύ τους, σε μια *διαδιεργασική επικοινωνία*.

Μια δημοφιλής τεχνική που χρησιμοποιείται στην επικοινωνία μεταξύ υπολογιστών είναι το μοντέλο client/server (πελάτη/διακομιστή ή πελάτη/εξυπηρετητή). Ο πελάτης (client) επιτελεί τις βασικές διεργασίες και τις αποστέλλει ως αίτημα, μέσω ενός hotspot (ενδιάμεσου σημείου), στο διακομιστή (server) ο οποίος περιμένει την αίτηση του client και την ικανοποιεί, προαιρετικά στέλνοντας πίσω μια απάντηση. Οι υπολογιστές, μαζί με αυτό το hotspot, συναποτελούν ένα *δίκτυο*.

Προκύπτει λοιπόν η ανάγκη ταυτοποίησης των υπολογιστών στο δίκτυο για να αναγνωρίσουν και να επικοινωνήσουν μεταξύ τους. Η ταυτοποίηση του καθενός γίνεται με μια διεύθυνση που ονομάζεται IP (από το ακρωνύμιο Internet Protocol). Υπάρχουν οι IPv4 και, λόγω της γεωμετρικής αύξησης των συσκευών που συνδέονται με το Διαδίκτυο και της επικείμενης εξάντλησης των IPv4, οι IPv6 διευθύνσεις. Οι IPv4 διευθύνσεις, που θα χρησιμοποιηθούν εδώ, είναι των 4 bytes αλλά αυτά συνήθως αναπαριστώνται ομαδοποιημένα στο δεκαδικό σύστημα, όπως για παράδειγμα *192.168.1.8*. Η ανάθεση μιας IP σε μια συσκευή μπορεί να γίνει είτε χειροκίνητα, επιλέγοντας μια ίδια στατική διεύθυνση που θα ανατίθεται στη συσκευή με κάθε εκκίνησή της είτε αυτόματα χρησιμοποιώντας πρωτόκολλο DHCP (Dynamic Host Configuration Protocol) και λαμβάνοντας μια από τις διαθέσιμες διευθύνσεις του δικτύου. Στην παρούσα διπλωματική εργασία θα οριστούν στατικές IP μιας και τα δεδομένα δε στέλνονται τυχαία αλλά στοχευμένα, από τον client στον server, οι οποίοι πρέπει να μην αλλάζουν διευθύνσεις (Comer, 2014).

Επειδή οι δύο αυτοί υπολογιστές τρέχουν εφαρμογές που πρέπει να συνεργαστούν, για την ανταλλαγή των δεδομένων απαιτείται κι άλλο ένα στοιχείο, που λέγεται *θύρα ή πόρτα (port)*. Η πόρτα είναι ένα *σημείο επικοινωνίας* μεταξύ δύο διεργασιών στο δίκτυο. Ο διακομιστής, έχοντας μια διεύθυνση IP, *"ακούει"* σε μια πόρτα για τα δεδομένα εφαρμογής του εκάστοτε πελάτη, ο οποίος με τη σειρά του τα μεταδίδει από μια δική του πόρτα. Οι πόρτες σε έναν υπολογιστή αριθμούνται από 1 έως 65535 γιατί για την αποθήκευσή του αριθμού αυτού χρησιμοποιείται μεταβλητή μη προσεσημασμένου ακέραιου 16 bit.

Ορισμένες πόρτες έχουν συγκεκριμένες ιδιότητες και χαρακτηριστικά. Η χρήση τους ακολουθεί ορισμένες συμβάσεις, με άλλα λόγια κάποιες συμφωνίες που έχουν επικρατήσει να εφαρμόζονται γενικώς στο Διαδίκτυο για να επιτυγχάνεται η επικοινωνία. Αυτές οι συμβάσεις είναι τα *πρωτόκολλα* (protocols), τα οποία ακολουθούνται μεταξύ των οντοτήτων ώστε τα δεδομένα που ανταλλάσσουν να έχουν νόημα. Δεν είναι απαγορευμένο όμως, πέραν των περιορισμών που υφίστανται λόγω των εγγενών ιδιοτήτων τους, να χρησιμοποιείται οποιοσδήποτε έγκυρος αριθμός πόρτας αν απλώς τηρηθούν κάποιες αμοιβαίες συμβάσεις μεταξύ των δύο μερών. Για παράδειγμα η πόρτα 23 έχει επικρατήσει να χρησιμοποιείται για το πρωτόκολλο telnet που είναι μη κρυπτογραφημένη ανταλλαγή δεδομένων. Σε μια εφαρμογή όμως όπου ο server γνωρίζει τί δεδομένα να αναμένει, στην πόρτα του με αριθμό 23 είναι δυνατόν να οριστεί να φτάσει οποιοδήποτε είδος πακέτων κάποιου client.

Για να χρησιμοποιηθεί μια πόρτα από το 1 έως το 1024 το πρόγραμμα πρέπει να έχει αυξημένα δικαιώματα διαχειριστή στο λειτουργικό σύστημα στο οποίο τρέχει. Συνεχίζοντας την αρίθμηση, μέχρι το 32768 (2^{15}) είναι οι *κοινώς γνωστές πόρτες* (well known ports), με γνωστά πρωτόκολλα στη διεθνή διευθυνσιοδότηση. Από τον αριθμό αυτό και έπειτα, οι πόρτες θεωρούνται αναλώσιμες, διαθέσιμες στο λειτουργικό σύστημα να τις δεσμεύσει στην τύχη όποτε είναι αναγκαίο.

Ο συνδυασμός διεύθυνσης IP και πόρτας ενός υπολογιστή λέγεται *Socket*. Ένας ορισμός που μπορεί να δοθεί είναι ότι πρόκειται για ένα προγραμματιστικό μηχανισμό μέσω του οποίου το πρόγραμμα που τρέχει σε κάθε υπολογιστή συνδέεται με το λειτουργικό σύστημα, λαβαίνει τα πακέτα δεδομένων ή και στέλνει άλλα σε έναν άλλο υπολογιστή. Μόλις τα δεδομένα αποσταλούν, η σύνδεση τερματίζεται και η πόρτα απελευθερώνεται και από τις δύο πλευρές.

Κάθε τέτοια σύνδεση υπολογιστών είναι *μοναδική* σε όλο το Διαδίκτυο. Αυτό επιτυγχάνεται με τη μοναδικότητα των Sockets των επικοινωνούντων υπολογιστών, δηλαδή με IP και port του client, και με IP και port του server.

Η Python διεθέτει ένα module γι' αυτήν ακριβώς τη λειτουργία με το όνομα `socket`. Είναι διαθέσιμο σε όλες τις τελευταίες διανομές τύπου Unix, στα Windows και στο macOS. Μερικές συμπεριφορές του module ίσως είναι εξαρτώμενες από την πλατφόρμα καθώς πρόκειται για κλήσεις που γίνονται προς το Application Programming Interface (API) (τη διεπαφή της προγραμματιστικής εφαρμογής) του λειτουργικού συστήματος. Είναι μια σχεδόν αυτούσια μετάφραση των system calls (κλήσεων συστήματος) των Unix με την προσέγγιση της Python. Για παράδειγμα η συνάρτηση `socket()` επιστρέφει ένα αντικείμενο τύπου `socket` του οποίου οι μέθοδοι υλοποιούν τα αντίστοιχα system calls. Οι τύποι των παραμέτρων είναι υψηλού επιπέδου, για παράδειγμα η δέσμευση μεγέθους buffer στις λειτουργίες λήψης δεδομένων είναι αυτόματη ενώ στις λειτουργίες αποστολής είναι σιωπηρό (Lutz, 2006).

6.4 – ΠΑΡΑΛΛΗΛΟΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ ΚΑΙ ΝΗΜΑΤΑ

Η βιομηχανία παραγωγής υλικού, για την αντιμετώπιση της συνεχούς αύξησης της πυκνότητας των τρανζίστορ, αντί να κατασκευάζει ολοένα και πιο σύνθετους μονολιθικούς επεξεργαστές, άλλαξε κατεύθυνση με το να ενσωματώνει πολλούς, σχετικά απλούς, πλήρεις επεξεργαστές σε ένα ολοκληρωμένο κύκλωμα. Οι συνδυασμένοι με αυτόν τον τρόπο επεξεργαστές ονομάστηκαν *πυρήνες (cores)* και αυτό το νέο είδος κυκλώματος έλαβε τον όρο *πολυπύρηνο (multicore)*. Έτσι, οι παλαιοί συμβατικοί επεξεργαστές αποκαλούνται πια *μονοπύρηνοι (single core)*.

Οι πολυπύρηνοι επεξεργαστές όρισαν και μια νέα προσέγγιση στη συγγραφή προγραμμάτων, αυτή της παραλληλοποίησης. Ο φόρτος εργασίας πλέον θα διοχετευόταν ανάμεσα στους πυρήνες, οι οποίοι θα αναλάμβαναν, συγχρονισμένα ή μη, ένα κομμάτι της όλης εργασίας. Τα παράλληλα προγράμματα είναι εξαρχής συνταγμένα έχοντας ένα σειριακό μέρος, το οποίο διοχετεύει τα δεδομένα στους πυρήνες και τα συνδυάζει όταν τα λαμβάνει από αυτούς, κι ένα παράλληλο μέρος, το οποίο απευθύνεται μόνο στους πυρήνες που εκτελούν τον κώδικα που τους αναλογεί.

Ένα σειριακό πρόγραμμα, κατά την εκτέλεσή του, αποτελεί ένα στιγμιότυπο το οποίο για το λειτουργικό σύστημα είναι μία *διεργασία* και μπορεί να εκτελεστεί το πολύ σε ένα πυρήνα. Το ίδιο πρόγραμμα μπορεί να εκτελεστεί σε πολλά στιγμιότυπα, το καθένα με δική του στοίβα, δικό του μετρητή προγράμματος στον επεξεργαστή και δικό του χώρο στην κύρια μνήμη. Όμως, σε ένα πολυπύρηνο σύστημα δεν ενδιαφέρει η εκτέλεση πολλών στιγμιοτύπων ενός προγράμματος αλλά η εκμετάλλευση των πυρήνων, με το διαμοιρασμό μεταξύ τους του φόρτου εργασίας ενός προγράμματος. Κάθε παράλληλο πρόγραμμα χωρίζει την εκτέλεση του στιγμιοτύπου σε μικρές διεργασίες που αποκαλούνται *νήματα* (threads) και τρέχουν το καθένα σε κάποιο πυρήνα.

Το λειτουργικό σύστημα αναλαμβάνει να δώσει χρόνο εκτέλεσης σε κάθε νήμα το οποίο, για το χρόνο αυτό, χρησιμοποιεί τον πυρήνα του επεξεργαστή που του έχει ανατεθεί. Ο χρόνος αυτός λέγεται *χρονομερίδιο* (time slice). Η απόδοση αυτού του κβάντου χρόνου ανατίθεται στα νήματα με υψηλή τυχαιότητα, πάντως έτσι ώστε να αποφεύγεται ένα φαινόμενο που λέγεται starvation, κατά το οποίο κάποιο νήμα δε λαμβάνει ποτέ χρόνο εκτέλεσης. Ανακύπτει λοιπόν η ανάγκη συγχρονισμού των νημάτων.

Τα νήματα συγχρονίζονται μεταξύ τους με διάφορες μεθόδους. Ενδεικτικά αναφέρονται οι εξής:

1. **Σηματοφορείς (semaphores)**. Χρησιμοποιούνται κοινές (global) μεταβλητές που λέγονται σηματοφορείς οι οποίοι ελέγχουν την πρόσβαση σε κάποιο κοινό πόρο ή στο κρίσιμο τμήμα ενός προγράμματος από τα διάφορα νήματα. Τα νήματα, κατά το χρόνο εκτέλεσής τους, αλλάζουν και την τιμή ενός σηματοφορέα ώστε τα υπόλοιπα να αλλάζουν τη συμπεριφορά τους ανάλογα με την τιμή αυτή.
2. **Σύγκριση και ανταλλαγή (compare and swap).** Η τεχνική αυτή συνίσταται στη σύγκριση

της τιμής μιας αρχικοποιημένης μεταβλητής με μια τιμή που επιθυμούμε να ελέγξουμε. Το νήμα που βρίσκει ισότητα εκτελείται, και αλλάζει την τιμή της μεταβλητής. Επίσης απαντάται σε κώδικα με κλειδωμένο το κρίσιμο τμήμα του με μια τέτοια μεταβλητή.

3. **Αμοιβαίος αποκλεισμός (mutual exclusion ή mutex).** Το παράλληλο πρόγραμμα εδώ έχει ένα κρίσιμο τμήμα[5]. Πρόκειται για ένα κομμάτι κώδικα στο οποίο δύο νήματα δε μπορούν να έχουν πρόσβαση σε αυτό κατά το ίδιο κβάντο χρόνου. Αυτό επιτυγχάνεται με τη χρήση μιας λογικής μεταβλητής που κλειδώνει το τμήμα αυτό. Το πρώτο νήμα που συναντάει το κρίσιμο τμήμα ελεύθερο, κλειδώνει τη μεταβλητή και εκτελεί τον κώδικα στο τμήμα αυτό. Αν το χρονομερίδιο του νήματος τελειώσει, κάποιο άλλο νήμα δε θα μπορεί να εισέλθει στο κρίσιμο τμήμα μιας και η λογική μεταβλητή είναι κλειδωμένη.
4. **Μεταβλητές συνθήκης (condition variables).** Οι μεταβλητές αυτές κι εδώ καταλαμβάνονται από κάποιο νήμα και χρησιμοποιούνται για το κλείδωμα κρίσιμου τμήματος προγράμματος. Τα υπόλοιπα νήματα εισέρχονται σε μια δομή ουράς ώστε να τηρείται μια σειρά προτεραιότητας. Μια μεταβλητή συνθήκης έχει ιδιότητες που καθορίζουν τα χαρακτηριστικά της συνθήκης για το κρίσιμο τμήμα.
5. **Σήματα (signaling).** Ο συγχρονισμός εδώ πραγματοποιείται με σήματα μεταξύ νημάτων. Κάποιο νήμα που βρίσκεται σε κατάσταση αναμονής, ενεργοποιείται μόλις λαμβάνει ένα σήμα (πρόκειται για σήμα του λειτουργικού συστήματος) από κάποιο άλλο νήμα ή νήματα. Αυτή η τεχνική μπορεί να χρησιμοποιηθεί σε συνδυασμό με κάποια άλλη, για παράδειγμα το ενεργοποιημένο σήμα μπορεί να πάρει τον έλεγχο κάποιας μεταβλητής κλειδώματος.
6. **Κλείδωμα τοποθεσίας (space location lock).** Για την αποτροπή επεξεργασίας δεδομένων, τίθεται κλείδωμα απευθείας σε bytes δεδομένων. Το κλείδωμα δεν είναι αναγκαίο να δημιουργηθεί από το εκτελούμενο πρόγραμμα και δεν επηρεάζει τα δεδομένα που προστατεύει.
7. **Κλείδωμα αντικειμένου (object lock).** Η μέθοδος αυτή εμπεριέχει το στοιχείο της αντικειμενοστρέφειας καθώς το κλείδωμα αντιστοιχεί σε αντικείμενο. Όταν ένα νήμα καλεί μια μέθοδο που χρησιμοποιείται για το συγχρονισμό, αυτομάτως αποκτά τον έλεγχο κλειδώματος στο αντίστοιχο αντικείμενο το οποίο απελευθερώνει όταν η μέθοδος επιστρέφει (Pacheco, 2011).

Ο συγχρονισμός των νημάτων στην παρούσα διπλωματική εργασία δε συνίσταται στον περιορισμό πρόσβασης αυτών στην ίδια μνήμη, αλλά σε μια προσέγγιση σύγκρισης τιμών μεταβλητών. Η εγγενής ιδιότητα των νημάτων να μειώνουν το χρόνο εκτέλεσης ενός προγράμματος ακριβώς για το λόγο επειδή εκτελούνται παράλληλα, είναι ένα ευπρόσδεκτο πλεονέκτημα.

Στο υλικό του Raspberry Pi 4, ο επεξεργαστής του διαθέτει τέσσερις πυρήνες με ισάριθμα νήματα. Στο πρόγραμμά μας χρησιμοποιούμε συνολικά τρία, με τον κώδικα σε Python να πραγματοποιεί το συγχρονισμό τους.

Ένα νήμα αναλαμβάνει την αποστολή του κρυπτογραφημένου αρχείου βίντεο. Επειδή αρχικά δεν υπάρχει κανένα τέτοιο αρχείο, το νήμα αναμένει ένα κρυπτογραφημένο αρχείο να δημιουργηθεί. Ένα δεύτερο νήμα έχει αναλάβει να κρυπτογραφήσει αρχείο βίντεο το οποίο αρχικά δεν υπάρχει οπότε κι αυτό αναμένει. Το τρίτο νήμα έχει ως λειτουργία τη βιντεοσκόπηση και δημιουργία τελικώς του αρχείου το οποίο εκκινεί τη διαδικασία και για τα τρία.

Με την εκκίνηση του νήματος της βιντεοσκόπησης, δημιουργείται ένα αρχείο βίντεο προκαθορισμένης διάρκειας, το οποίο αποκαλείται κομμάτι (chunk). Ύστερα, ο έλεγχος του προγράμματος περνάει στο κρυπτογραφικό νήμα που αναμένει το αρχείο βίντεο. Με την κρυπτογράφηση του αρχείου, το νήμα αυτό τελειώνει τη λειτουργία του και μεταβιβάζει τον έλεγχο στο νήμα αποστολής που τελικώς στέλνει το αρχείο. Μετά από αυτές τις τρεις λειτουργίες, όλα τα νήματα καταστρέφονται. Για το επόμενο κομμάτι (chunk) αλλά και για τα τυχόν επόμενα, θα δημιουργηθούν τρία νέα νήματα, τα οποία θα λειτουργήσουν και θα καταστραφούν με τον ίδιο τρόπο.

Ο συγχρονισμός τους πραγματοποιείται με μεταβλητές ως μετρητές. Οι μετρητές

5 Μέσα στο παράλληλο πρόγραμμα υπάρχει ένα μέρος στον κώδικα το οποίο ονομάζεται κρίσιμο τμήμα το οποίο εκτελείται μόνο από ένα νήμα κάθε φορά, υπό συνθήκες. Ο όρος κρίσιμο τμήμα χρησιμοποιήθηκε για πρώτη φορά από τον Edsger Dijkstra το 1965 στο ακαδημαϊκό κείμενο “Solution of a problem in concurrent programming control”.

αρχικοποιούνται με μηδενική τιμή. Οι τιμές αυτές εκτός από το συγχρονισμό των νημάτων χρησιμοποιούνται και στην ονομασία των αρχείων που θα βιντεοσκοπηθούν και θα κρυπτογραφηθούν, αντίστοιχα. Μόλις το νήμα που βιντεοσκοπεί δημιουργήσει το αρχείο, αυξάνει το μετρητή βιντεοσκόπησης. Το νήμα που κρυπτογραφεί συγκρίνει το δικό του μετρητή με αυτόν του νήματος βιντεοσκόπησης οπότε είναι αδύνατο να προπορευτεί προσπαθώντας να κρυπτογραφήσει αρχείο που δεν υπάρχει ακόμη. Με τον ίδιο τρόπο το νήμα που αποστέλλει συγκρίνει το μετρητή του με αυτόν του νήματος κρυπτογράφησης και συνεπώς αναμένει με τη σειρά του να δημιουργηθεί το κρυπτογραφημένο αρχείο κάθε φορά πριν προχωρήσει.

6.5 – RC.LOCAL ΚΑΙ ΑΥΤΟΜΑΤΟΠΟΙΗΜΕΝΗ ΕΚΚΙΝΗΣΗ

Υπάρχει κι ένα κομμάτι της όλης διαδικασίας που την αυτοματοποιεί. Για να αναγκαστούν τα Python αρχεία στον server και στον client να ξεκινήσουν αυτόματα πρέπει να τροποποιηθεί το αρχείο που βρίσκεται στη διαδρομή `/etc/rc.local`. Είναι ένα αρχείο με εντολές κελύφους, που τις τρέχει με την εκκίνηση του λειτουργικού συστήματος, καθώς και χρήσιμα σχόλια που τις επεξηγούν. Σημειώνεται ότι το αρχείο αυτό ξεκινάει στο λειτουργικό σύστημα πριν την έναρξη του γραφικού περιβάλλοντος, οπότε γενικά περιέχει εντολής που δεν αφορούν το GUI.

Εδώ, θα προστεθεί η εντολή `python server01.py` και `python client01.py` στον server και στον client αντίστοιχα ώστε να τρέξουν αυτοματοποιημένα τα αρχεία που τους εκκινούν. Πρέπει όμως να διαβεβαιωθεί ότι κάθε μηχάνημα όχι μόνο θα έχει συνδεθεί στο router και θα έχει αποδοθεί IP αλλά θα είναι και η σωστή που απαιτείται. Συνεπώς στον client προστείθεται όλο το snippet:

```
my_ip=$(hostname –I | awk '{print $1}')
ip_check="192.168.1.31"
if [ "$my_ip" == "$ip_check" ]; then
        python client01.py
fi
```

Σε αυτό το κομμάτι εντολών συμβαίνουν τα εξής. Αποθηκεύεται στη μεταβλητή `my_IP` την αποτίμηση της φράσης στην παρένθεση. Το `hostname –I` δίνει την IP διεύθυνση σε διάφορες μορφές (τετράδα bytes, IPV4, IPv6) από τις οποίες πρέπει να επιλεχθεί μόνο η πρώτη. Έτσι, σωληνώνεται το αποτέλεσμα της εντολής σε `awk`[6] που επιλέγει την πρώτη από τις τιμές. Στη μεταβλητή `ip_check` τίθεται η IP που απαιτείται να έχει το Raspberry Pi ώστε να έχει νόημα το τρέξιμο του αρχείου.

Αμέσως μετά τοποθετείται ένας έλεγχος `if` που συγκρίνει τις συμβολοσειρές αποθηκευμένες στις μεταβλητές `my_IP` και `ip_check`. Αν είναι ίδιες, σημαίνει ότι το Raspberry Pi έλαβε την επιθυμητή IP και είναι πλέον δόκιμο να τρέξει το αρχείο `client01.py`. Αντίστοιχα μεταβάλλεται το ίδιο αρχείο και στον server.

Ήταν δυνατόν να τεθεί κάποια άλλη εντολή ελέγχου όπως `while`, `do – while` ώστε να γίνεται συνεχώς έλεγχος για τη σωστή IP. Αυτό όμως δεν έχει νόημα. Υποτίθεται ότι χρησιμοποιείται μια συγκεκριμένη static IP. Αν το Raspberry Pi για κάποιο λόγο αλλάζει συνεχώς IP ή λάβει μια απρόβλεπτη αυτό ίσως σημαίνει ότι ίσως κάποια μη εξουσιοδοτημένη οντότητα προσπαθεί να υποδυθεί το πραγματικό router και να εξαναγκάσει τα δύο Raspberry Pi να συνδεθούν σε αυτό. Έτσι, με την `if` τοποθετείται εδώ ένα επιπρόσθετο μέτρο ασφαλείας που προστατεύει έως ένα βαθμό από διαρροή δεδομένων σε κάποιο επίδοξο επιτιθέμενο στο σύστημα.

6 Τα αρχικά awk προκύπτουν απότα επώνυμα των δημιουργών του, Alfred **A**ho, Peter **W**einberger και Brian **K**ernighan. Η awk είναι εφαρμογή που δημιουργήθηκε το 1977 και πραγματοποιεί ενέργειες σε αρχεία, βάσεις δεδομένων, ευρετήρια κλπ όταν η είσοδος είναι ένα επιλεγμένο μοτίβο. Ο όρος awk αναφέρεται στο πρόγραμμα αλλά και στη αντίστοιχη γλώσσα.

ΚΕΦΑΛΑΙΟ 7
ΠΕΡΙΓΡΑΦΗ ΥΛΙΚΟΥ

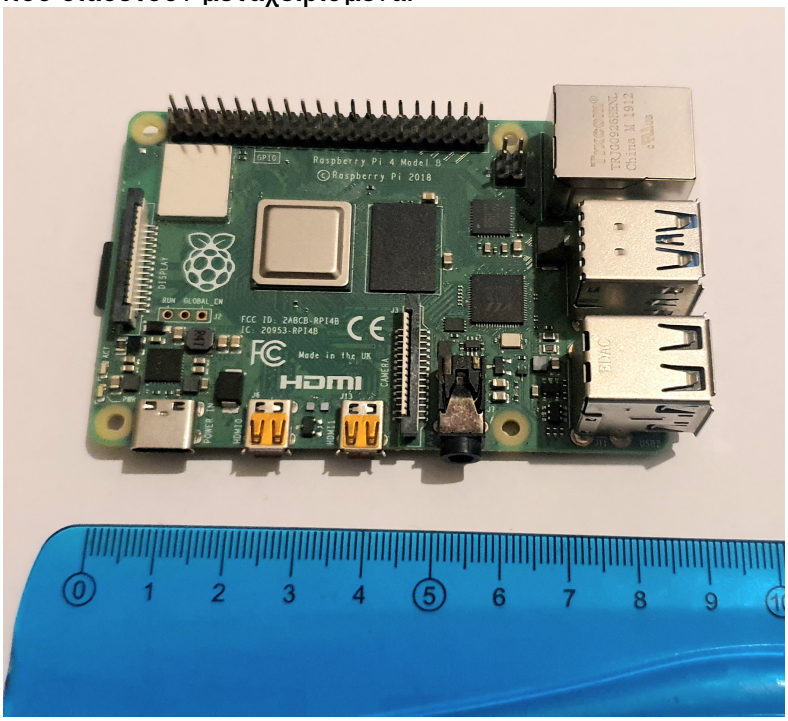
7.1 - ΓΕΝΙΚΑ

Σε αυτό το κεφάλαιο θα παρουσιαστούν τα συγκεκριμένα κομμάτια hardware που χρησιμοποιήθηκαν για την υλοποίηση. Εκτός από τα Raspberry Pi, τις τροφοδοσίες τους και την κάμερα, χρησιμοποιήθηκαν κι άλλα εξαρτήματα. Ξεκινώντας από το μηδέν, θα κατασκευαστεί ένα ολοκληρωμένο σύστημα που θα περιλαμβάνει ό,τι απαιτείται για να καταγράφει, να κρυπτογραφεί και να αποστέλλει δεδομένα με ένα δεύτερο σημείο όπου επιβεβαιώνεται η λειτουργία του.

7.2 - ΒΑΣΙΚΑ ΥΛΙΚΑ

RASPBERRY PI 4

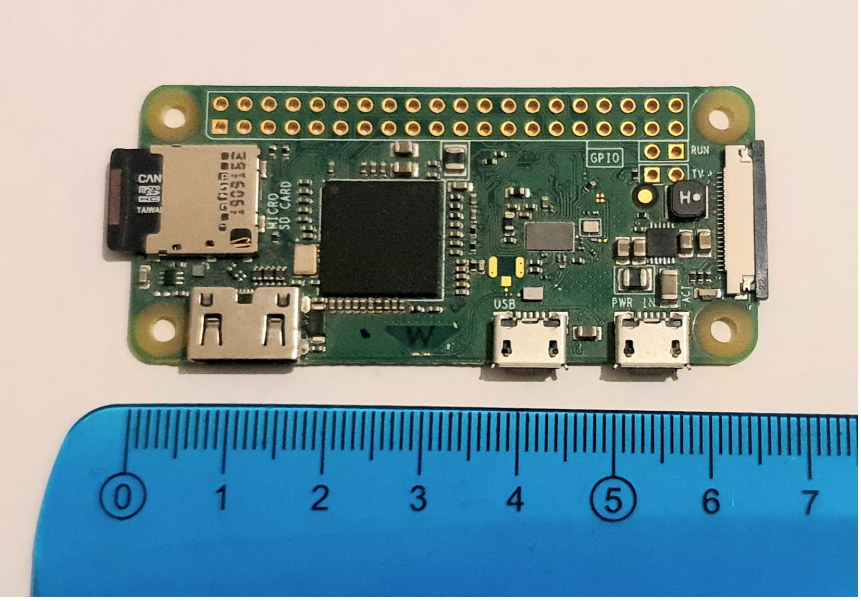
Ως client χρησιμοποιήθηκε ένα Raspberry Pi 4 – model B, revision 2, η έκδοση με 1 GiB RAM. Το συγκεκριμένο μοντέλο έχει σταματήσει την κυκλοφορία του. Δεν παράγονται νέα κομμάτια και είναι πολύ δύσκολο να ανευρεθεί σε καταστήματα πώλησης εκτός από ορισμένα διάσπαρτα σημεία που διαθέτουν μεταχειρισμένα.



Εικόνα 7.1
Raspberry Pi 4

RASPBERRY Pi ZERO W

Το Raspberry Pi Zero W είναι μια spinoff έκδοση του Raspberry Pi 1 που κατασκευάστηκε αρκετά μετά, ως ένα σύστημα μικρότερου μεγέθους και κόστους. Το παρόν είναι το Raspberry Pi Zero W (διαθέτει δυνατότητα WiFi), revision 2 και χρησιμοποιήθηκε ως server. Η παραγωγή αυτού του μοντέλου έχει επίσης σταματήσει εδώ και αρκετό καιρό.



Εικόνα 7.2
Raspberry Pi Zero W

ROUTER

Για την επικοινωνία του client και του server μεταξύ τους χρειάζεται ένα hotspot. Αυτό είναι το router ZTE Home Gateway ZXHN H267N (έκδοση Σεπτεμβρίου 2016). Σημειώνεται ότι δεν απαιτείται να έχει πρόσβαση στο Διαδίκτυο. Αυτό που θα κάνει είναι να αποδώσει IP (και συγκεκριμένα στατικές) στα δύο Raspberry Pi και να αναλάβει το διαμεσολαβητή ανταλλαγής δεδομένων ανάμεσά τους.

Εικόνα 7.3
Το router / hotspot ZTE Home Gateway ZXHN H267N

ΚΑΜΕΡΑ

Χρησιμοποιήθηκε μια απλή κάμερα που συνδέθηκε σε μια από τις USB θύρες του Raspberry Pi 4. Η συγκεκριμένη είναι η Crypto Joker Webcam με τύπο διασύνδεσης USB 2.0 και μέγιστη native ανάλυση 640 x 480 pixels. Το εργοστασιακό της βάρος είναι 93,5 γραμμάρια. Καταγράφει στα 30 frames per second, ενώ μπορεί να αυξήσει την ανάλυση καταγραφής στα

1280 x 960 pixels, αλλά πέφτοντας στα 15 frames per second. Ο αισθητήτας της είναι 1/6 CMOS VGA με βάθος χρωμάτων RGB 24 bit, με αυτόματη εστίαση ξεκινώντας από τα 80 mm έως το άπειρο. Επίσης διαθέτει και ενσωματωμένο μικρόφωνο.

Εικόνα 7.4
Crypto Joker Webcam

ΤΡΟΦΟΔΟΣΙΑ

Για την τροφοδοσία του Raspberry Pi 4 (client) χρησιμοποιήθηκε power bank της GoodRam χωρητικότητας 5000 mAh. Πρόκειται για μπαταρία ιόντων λιθίου με εργοστασιακή διάρκεια ζωής 500 κύκλων φόρτισης. Υπάρχουν δύο θύρες των 5 V στο 1,5 A, για φόρτιση και αποφόρτιση, και η επιλογή λειτουργίας γίνεται αυτόματα είτε με την σύνδεση πηγής ενέργειας είτε συσκευής που απαιτεί ρεύμα. Το εργοστασιακό της βάρος είναι 125 γραμμάρια.

Εικόνα 7.5
Power bank GoodRam 5000 mAh

ΒΑΡΟΣ ΚΑΤΑΣΚΕΥΗΣ

Όλα τα εξαρτήματα του server έχουν σημαντικό και μη αμελητέο βάρος. Στο συνδυασμό απαιτείται κι ένα καλώδιο για τη σύνδεση του Raspberry Pi 4 με την power bank. Όλα τα παραπάνω ζυγίζουν 276 γραμμάρια.

Εικόνα 7.6
Ζύγιση κατασκευής client

DRONE PAYLOAD - ΦΟΡΤΙΟ

Κάθε drone χαρακτηρίζεται από την ικανότητα για φορτίο (payload). Είναι το συνολικό βάρος όλων των εξαρτημάτων του που δε χρησιμεύουν για πτήση. Επηρεάζει την

ευελιξία του, την ταχύτητα, την κίνηση και τη διάρκειά της ‘ είναι φανερό ότι όσο πιο μεγάλο βάρος μεταφέρει τόσο πιο δυσχερής γίνεται ο χειρισμός.

Το payload είναι ένα μέγεθος που έρχεται από τον κατασκευαστή και εξαρτάται από την εργοστασιακή ικανότητα του drone. Ένα που χρησιμοποιείται σε αγώνες, για παράδειγμα, θα έχει μηδενικό (ή ελάχιστο) payload καθώς δημιουργήθηκε για ταχύτητα και ευελιξία ενώ ένα κατασκοπευτικό ή πολεμικό θα έχει ιδιαίτερα αυξημένη αυτή την ικανότητα αφού φτιάχτηκε για να μεταφέρει εξοπλισμό παρακολούθησης ή και πυρομαχικά.

Υπάρχει κι ένα άλλο μέγεθος που αφορά συγκεκριμένα τα ιπτάμενα drone, το MTOW (Maximum Take Off Weight) το οποίο ουσιαστικό είναι το μικτό βάρος, αλλά αναφέρεται ως το *μέγιστο βάρος απογείωσης*. Αν υπερκαλυφθεί αυτό το βάρος τότε το drone δε μπορεί να απογειωθεί καθόλου.

7.3 – ΚΟΣΤΟΣ ΥΛΟΠΟΙΗΣΗΣ

Οι παρακάτω τιμές υπολογίζονται όπως αγοράστηκαν τη στιγμή της κυκλοφορίας τους ή ως μεταχειρισμένα*. Έχουν προστεθεί αντάπτορες HDMI καθώς και οθόνη ώστε να συνδεθούν τα Raspberry Pi για την ανάπτυξη κώδικα και τις διάφορες ρυθμίσεις σε αυτά, όπως και για την αναπαραγωγή βίντεο και επιβεβαίωση της υλοποίησης. Επίσης το Zero χρειάζεται έναν αντάπτορα ρεύματος για να συνδεθεί σε πρίζα (μιας και θεωρείται σταθμός εδάφους, είναι σταθερός) και ένας αντάπτορας για data ο οποίος θα συνδεθεί στη δεύτερη θύρα microUSB που διαθέτει για την προσθήκη hub με πληκτρολόγιο και ποντίκι.

Με το παρακάτω συνολικό ποσό δίνεται μια ολοκληρωμένη λύση της υλοποίησης, κατασκευασμένη εκ του μηδενός, χωρίς προϋπάρχοντα υλικά.

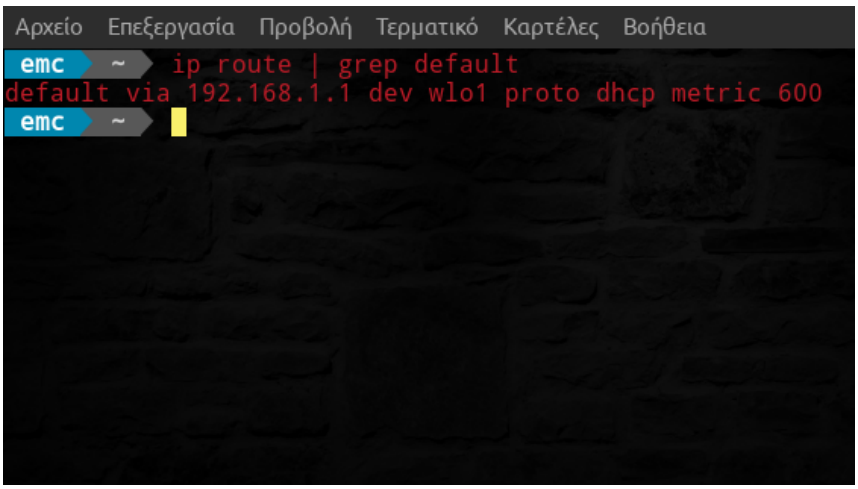
Είδος	Κόστος (€)
Raspberry Pi Zero W *	9.00
microUSB type-A charger (RPi0)	4.00
USB hub	2.50
2 x HDMI αντάπτορες	5.00
2 x HDMI καλώδια	10.00
Raspberry Pi 4 – model B *	45.00
Raspberry Pi 4 τροφοδοσία	10.00
Crypto Joker Webcam	12.50
Power Bank GoodRam 5000 mAh	8.50
USB type-C cable	2.50
Οθόνη με καλώδιο τροφοδοσίας *	125.00
Πληκτρολόγιο & ποντίκι	20.00
ΣΥΝΟΛΟ	**254.00**

ΚΕΦΑΛΑΙΟ 8
ΑΝΑΛΥΣΗ ΚΩΔΙΚΑ

8.1 – ΣΗΜΕΙΟ ΠΡΟΣΒΑΣΗΣ – ACCESS POINT

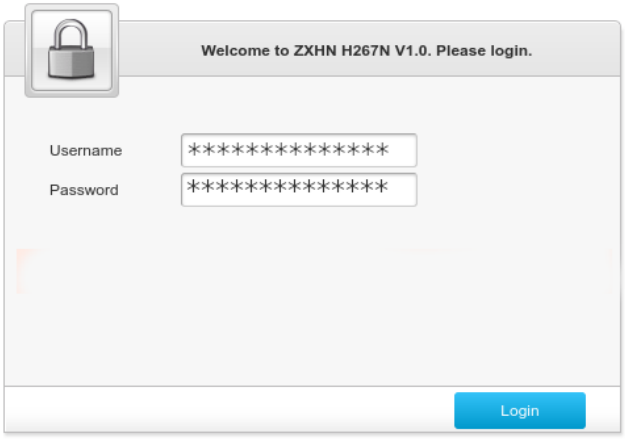
Ως σημείο πρόσβασης χρησιμοποιήθηκε το προαναφερθέν απλό router. Δεν είναι συνδεδεμένο στο Διαδίκτυο γιατί δε χρειάζεται. Ο ρόλος του είναι να αποδώσει IP στις συσκευές που είναι συνδεδεμένες σε αυτό και να αναλάβει να ανταλλάξει τα δεδομένα μεταξύ τους.

Το router αυτό καθ' αυτό έχει προρυθμισμένη μια SSID (Service Set IDentifier). Πρόκειται ουσιαστικά για το όνομα που μεταδίδεται συνεχώς εντός της εμβέλειάς του και χρησιμεύει ώστε να είναι δυνατή η ανεύρεσή του και η σύνδεση σε αυτό, με τον εργοστασιακό του κωδικό. Αμέσως μετά, ανακτάται μέσω τερματικού η IP του, μέσω της εντολής `ip`. Σε αυτή θα προστεθεί και το αντικείμενο `route` ώστε να εμφανιστεί όλο το routing table του υπολογιστή, και μετά σωληνώνεται με τη *regular expression* `default` (ώστε να εμφανίσει μόνο τις γραμμές με τη default gateway, την προεπιλεγμένη πύλη, το router). Άρα λοιπόν:



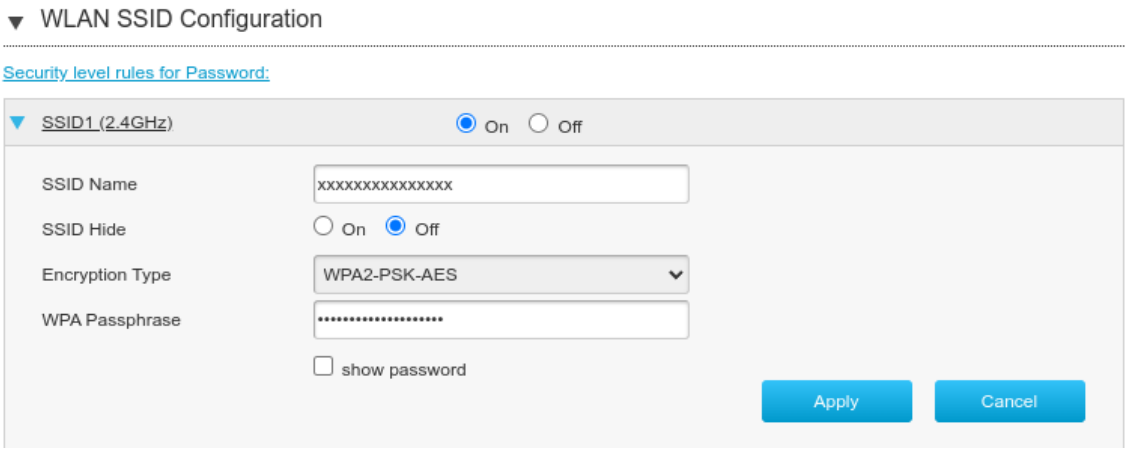
Εικόνα 8.1
Εύρεση IP του router

Αφού βρέθηκε η IP του router, εισάγεται σε οποιονδήποτε browser και έτσι εμφανίζεται ένα παράθυρο διαλόγου εισόδου στις ρυθμίσεις του router.



Εικόνα 8.2
Είσοδος στις ρυθμίσεις του router

Από τις Ρυθμίσεις SSID αλλάζεται άμεσα η SSID και ο κωδικός πρόσβασης στο WiFi. Τα εργοστασιακά στοιχεία είναι εξαιρετικά ευάλωτα σε επιθέσεις. Ο εργοστασιακός κωδικός για το WiFi, σε ορισμένα εξάγεται μέσω κάποιου αλγορίθμου από το όνομα της SSID (που είναι σε όλους φανερό). Ακόμη, κυκλοφορούν στο Διαδίκτυο πολλά text αρχεία που περιέχουν χιλιάδες ανακτημένους κωδικούς, εργοστασιακούς και μη, τους οποίους κάποιο πρόγραμμα μπορεί να δοκιμάσει με τη σειρά και σύντομα να αποκτήσει πρόσβαση στο router. Γι' αυτό είναι αναγκαίο όχι μόνο να αλλάξει ο εργοστασιακός κωδικός αλλά και να οριστεί ένας άλλος ισχυρότερος. Προτείνεται γενικά να χρησιμοποιούνται πάνω από 12 χαρακτήρες, οι οποίοι να είναι γράμματα, κεφαλαία και πεζά, τουλάχιστον δύο αριθμοί και πάνω από δύο σύμβολα, χωρίς να περιλαμβάνεται μέσα κάποια κοινή λέξη ή προφανής προσωπική πληροφορία.



Εικόνα 8.3
Αλλαγή προεπιλεγμένου κωδικού του WiFi του router

Αφού αλλάχθηκε και ο κωδικός πρόσβασης του WiFi του router, παρατηρείται ότι ο υπολογιστής με τον οποίο έγιναν οι ρυθμίσεις στο router δεν έχει πλέον πρόσβαση σε αυτό, καθ' όσον χρησιμοποιεί τα παλιά credentials τα οποία μόλις αντικαταστάθηκαν. Για να είναι δυνατή η επανασύνδεση πρέπει να χρησιμοποιηθούν τα νέα.

Τώρα πρέπει να ρυθμιστούν τα δύο Raspberry Pi, ένα εκ των οποίων θα είναι ο client και το άλλο ο server. Είναι αναγκαίο να οριστούν στατικές IP γιατί ο κώδικας της Διπλωματικής εργασίας είναι γραμμένος έτσι ώστε να τις εκμεταλλεύεται, αφού δε θα λειτουργήσει αν στέλνουμε αδιακρίτως δεδομένα σε τυχαίες IP. Η υλοποίηση πρέπει να είναι σαφώς ορισμένη ώστε να επικοινωνούν δύο αυτές συσκευές με συνέπεια. Η απόδοση των στατικών IP γίνεται με πολλούς τρόπους. Εδώ θα οριστούν παραπάνω από ένας χωρίς να υπάρχει διένεξη. Από τη στιγμή που κάποιος τρόπος ορίζει μια συγκεκριμένη στατική IP, ένας άλλος τρόπος που ορίζει την ίδια συγκεκριμένη στατική IP απλώς την επιβεβαιώνει και τη χρησιμοποιεί.

Κάθε συσκευή χαρακτηρίζεται από διευθύνσεις MAC. Αυτή είναι μια διεύθυνση υλικού, είναι ένας μοναδικός αριθμός που αντιστοιχεί σε κάθε κάρτα wireless interface μιας συσκευής (Ethernet, Bluetooth, Wifi κλπ) και για το module δικτύου είναι της μορφής xx.xx.xx.xx.xx.xx, έξι ζευγάρια δεκαεξαδικών ψηφίων. Επειδή τα δύο Raspberry Pi συνδέονται στο router με Wifi, το router μόλις παρατηρήσει μια συσκευή να προσπαθεί να συνδεθεί σε αυτό, από προεπιλογή της αποδίδει μια τυχαία IP διεύθυνση από τις διαθέσιμες. Αυτό είναι το πρωτόκολλο DHCP (Dynamic Host Configuration Protocol), το οποίο όμως πρέπει να παρακαμθεί αφού απαιτούνται στατικές IP και όχι αποδιδόμενες δυναμικά. Δηλαδή όταν το router παρατηρήσει μια συσκευή με μια συγκεκριμένη διεύθυνση MAC (που αντιστοιχεί στο interface δικτύου) πρέπει να της αποδώσει IP ορισμένη στατικά, όχι τυχαία.

Όταν είναι κάποιο Raspberry Pi συδεδεμένο ήδη, με τα credentials ασφαλείας του router από τις Ρυθμίσεις του που αφορούν τις συνδεδεμένες συσκευές του τοπικού δικτύου το βρίσκουμε και ορίζουμε στατική IP. Συγκεκριμένα, για τον client θα είναι η 192.168.1.30 και για τον

server η 192.168.31.

Ένας άλλος τρόπος είναι η τροποποίηση του αρχείου `/etc/dhcpcd.conf`. Για παράδειγμα, στο τέλος του αντίστοιχου αρχείου στον client θέτοντας τις τελευταίες γραμμές ως:

```
interface wlan0
static ip_address = 192.168.0.30/24
static routers = 192.168.1.1
static domain_name_servers = 192.168.1.1
```

Είναι ακόμη ένας τρόπος ώστε το Raspberry Pi να ζητάει, σε οποιοδήποτε router (που έχει IP 192.168.1.1) προσπαθεί να συνδεθεί, την IP 192.168.1.30, όταν προσπαθεί να συνδεθεί με Wifi (δηλαδή το interface wlan0). Η διεύθυνση αυτή δε θα δοθεί στο Raspberry Pi μόνο αν έχει καταληφθεί ήδη από μια άλλη συσκευή. Το router δηλαδή δε θα αποσυνδέσει την άλλη συσκευή για να δώσει στο Raspberry Pi τη διεύθυνση που επιθυμεί.

Μετά από αυτά τα βήματα δε χρειάζεται να ρυθμιστεί τίποτα άλλο στο router.

8.2 - RASPBERRY PI 4 – CLIENT

Θα χρησιμοποιηθεί το Raspberry Pi 4 ως client αφού θα αναλάβει την πιο απαιτητική εργασία. Χρειάζεται να κινηματογραφήσει βίντεο, να το κρυπτογραφήσει και να το αποστείλει στον server που αναμένει. Θα επιστρατευθούν threads γι' αυτό, όχι για να παραλληλοποιηθούν οι εργασίες αλλά για να λειτουργήσει μια άλλη ικανότητα των νημάτων, η *αναμονή*.

Όπως φαίνεται στον κώδικα του client που ακολουθεί, οι τρεις προαναφερθείσες εργασίες του εκτελούνται με τη σειρά, και δεν ξεκινάει η μια αν δεν τελειώσει η προηγούμενη. Στη βασική μέθοδο `main` που εκτελείται με την έναρξη του προγράμματος έχει τεθεί μετρητής για να γίνονται δύο επαναλήψεις ώστε να δειχθεί το ζητούμενο της υλοποίησης. Εξυπακούεται όμως ότι αν στην επανάληψη τεθεί `while True` ο client θα εκτελεί το πρόγραμμα για πάντα (στην πράξη μέχρι να τελειώσει η μπαταρία του συστήματος).

8.2.1 - ΚΩΔΙΚΑΣ CLIENT ΚΑΙ ΕΠΕΞΗΓΗΣΗ

```
import threading
import time
import sys

import socket
import os

import cv2
from cv2 import VideoWriter
from cv2 import VideoWriter_fourcc


from Crypto.Cipher import AES
from Crypto.Util.Padding import pad, unpad
from base64 import b64encode, b64decode

HOST = "192.168.0.31"
PORT = 50001

# The size of the buffer for each file we will send
BUFFER_SIZE = 4096
SEPARATOR = "<SEPARATOR>"

recordCount = 0
```

```
encryptCount = 0
sendCount = 0

# Key, if needed is padded with extra (or less) characters
key = "thisK3yW1llN3v3rB3F0und@@"
key = key.encode('UTF-8')
key = pad(key, AES.block_size)


#  ======================================================
#  M E T H O D S
#  ======================================================

def recWithCV(filename):
        # (0) is for the default camera
        webcam = cv2.VideoCapture(0)
        videoWidth = 640
        videoHeight = 480

        # open output video file stream
        # VideoWriter(filename, codec, framesPerSecond, (dimensions))
        video = VideoWriter(filename, VideoWriter_fourcc(*'MP42'), 10.0,
(videoWidth, videoHeight))

        startingTime = time.time()

        # camera frames loop
        while True:
                # get the frame from the webcam
                # stream_ok -> variable for working webcam (or not)
                stream_ok, frame = webcam.read()

                # if webcam stream is ok
                if stream_ok:
                        # display current frame in pop -up window
                        cv2.imshow('Recording...', frame)

                        # write frame to the video file
                        video.write(frame)
                        endingTime = time.time()
                        if endingTime - startingTime > 10.0:
                                break

        cv2.destroyAllWindows()

                # release web camera stream
        webcam.release()
                # release video output file stream
        video.release()



def record():
        print("\n==================================\nRECORDING THREAD
INITIATED")
        global recordCount
        recordFile = "emc_video{0}.avi".format(recordCount)
        recWithCV(recordFile)
        print("Ten seconds have been recorded to ", recordFile)
        recordCount = recordCount + 1
        print("Recording thread ended\n")
        time.sleep(2)
```

```
def encrypt():
        print("\n====================\nENCRYPTING THREAD INITIATED")
        global encryptCount
        global recordCount
        global key
        while encryptCount<recordCount:
                toBeEncrypted = "emc_video" + str(encryptCount) + ".avi"
                encryptFile = "encrypted + str(encryptCount)"
                with open(toBeEncrypted, 'rb') as entry:
                        data = entry.read()

                        # CFB mode of AES
                        cipher = AES.new(key, AES.MODE_CFB)
                        cipherText = cipher.encrypt(pad(data,
AES.block_size))
                        iv = b64encode(cipher.iv).decode('UTF-8')

                        cipherText = b64encode(cipherText).decode('UTF-8')
                        to_write = iv + cipherText
                entry.close()
                with open(encryptFile + ".enc", 'w') as data:
                        data.write(to_write)
                data.close()

                print("The file ", encryptFile, " has been encrypted")
                encryptCount = encryptCount + 1
                print("Encrypting thread ended\n")


def send():
        print("\n====================\nSENDING THREAD INITIATED")
        global sendCount
        global HOST
        global PORT
        global SEPARATOR
        global BUFFER_SIZE

        while sendCount<encryptCount:
                sendFile = "encrypted" + str(sendCount) + ".enc"
                filesize = os.path.getsize(sendFile)

                # Creating the client socket and connecting to it
                s = socket.socket()
                print(f"[+] Connecting to {HOST}:{PORT}")
                s.connect((HOST, PORT))
                print("[+] CONNECTED!")
                s.send(f"{sendFile}{SEPARATOR}{filesize}".encode())
                # encode() function encodes the string we passed to UTF-8

                # Sending the file, opening as "read in binary"
                with open(sendFile, "rb") as f:
                        while True:
                                # Read bytes from the file
                                bytes_read = f.read(BUFFER_SIZE)
                                if not bytes_read:
                                        # File transmitted successfully
                                        break
                                s.sendall(bytes_read)
                s.close()
                print("  Sending is over.")
                print("File ", sendFile, " successfully sent to server")
                sendCount = sendCount + 1
                print("Sending thread ended\n")
```

```
#  ======================================================
#  M A I N   P R O G R A M
#  ======================================================
if __name__ == "__main__":
        print("--- The whole process starts here. ---")

        # Replace with while(True) to repeat forever
        countdown = 2
        while(countdown):
                send_thread = threading.Thread(target=send)
                encrypt_thread = threading.Thread(target=encrypt)
                record_thread = threading.Thread(target=record)

                record_thread.start()
                record_thread.join()

                encrypt_thread.start()
                encrypt_thread.join()

                send_thread.start()
                send_thread.join()

                countdown = countdown - 1
```

Τρία είναι τα threads, το `record_thread` (που κινηματογραφεί), το `encrypt_thread` (που κρυπτογραφεί το αρχείο βίντεο) και το `send_thread` (που αποστέλλει τα κρυπτογραφημένα δεδομένα).

Για τις ανάγκες της υλοποίησης έχει οριστεί ενδεικτικά ένα πεπερασμένο πλήθος επαναλήψεων με ένα μετρητή, `countdown = 2`. Αυτό σημαίνει ότι η διαδικασία στον client θα επαναληφθεί δύο φορές, για τις τιμές 2, και 1 της `countdown`. Όταν `countdown = 0` τότε η αποτίμηση `while(countdown)` μεταφράζεται ως False και η επανάληψη σταματάει.

Σε κάθε επανάληψη δημιουργούνται αντικείμενα του κάθε thread από τη βασική κλάση `threading` (που έχει αντικαταστήσει την παρωχημένη κλάση `thread`). Αμέσως μετά καλείται η ενσωματωμένη μέθοδος νημάτων, η `start()` ώστε να ξεκινήσει το κάθε νήμα να τρέχει. Αμέσως μετά, καλείται η επίσης ενσωματωμένη μέθοδος νημάτων `join()` η οποία υπαγορεύει στη `main` ότι το παρόν νήμα που τρέχει πρέπει πρώτα να ολοκληρώσει την εργασία του και μετά να προχωρήσει η περαιτέρω εκτέλεση του κώδικα. Αυτή είναι η προαναφερθείσα αναμονή, που τίθεται σε όλα τα νήματα.

Άρα λοιπόν το `record_thread` θα ολοκληρώσει την εργασία του και μετά η `main` θα προχωρήσει την εκτέλεση, εκκινώντας το νήμα `encrypt_thread` που θα κρυπτογραφήσει. Μέχρι το τέλος της κρυπτογράφησης το νήμα `send_thread` αναμένει ' μετά η `main` θα του δώσει τον έλεγχο ξεκινώντας το, που θα τρέξει με όμοιο τρόπο. Μετά την εκτέλεση κάθε νήματος, αυτό καταστρέφεται. Ο μετρητής `countdown` μειώνεται κατά ένα και η διαδικασία επαναλαμβάνεται.

Ο αλγόριθμος του client θεωρείται σωληνωμένος (pipelined). Δηλαδή η έξοδος επεξεργασίας για μια ομάδα δεδομένων αποτελεί είσοδο για την επόμενη επεξεργασία τους. Κάθε ένα από τα τρία νήματα έχει ως στόχο μια μέθοδο, φαίνεται στο όρισμα κατασκευής του αντικειμένου του κάθε νήματος ως `target`. Η `record()` καλεί τη `recWithCV()` που επιτελεί την επιμέρους εργασία αυτής καθ' αυτής της κινηματογράφησης, χωρίς αυτό να διαταράσσει την ομαλή εκτέλεση του προγράμματος. Μέχρι να τελειώσει η βασική μέθοδος του νήματος `record_thread`, τα υπόλοιπα νήματα περιμένουν.

Είναι επίσης αναγκαίο γενικά οι μεταβλητές να είναι αφενός μεν global, αφετέρου δε οι μέθοδοι να έχουν πρόσβαση σε αυτές (οι τοπικές μεταβλητές έχουν εμβέλεια μόνο μέσα, και κατά τη διάρκεια εκτέλεσης μιας μεθόδου).

Πρέπει να αναφερθεί εδώ ότι επειδή η Python χρησιμοποιεί διερμηνευτή, είναι αναγκαίο οι μέθοδοι, οι μεταβλητές και τα λοιπά στοιχεία να ορίζονται *πριν* κληθούν οπουδήποτε στον κώδικα.

ΑΡΧΙΚΕΣ ΣΥΜΒΑΣΕΙΣ

Αρχικά, γίνονται τα απαραίτητα `import` ώστε να έχει ο διερμηνευτής τις βιβλιοθήκες στη διάθεσή του. Όπως προαναφέρθηκε στα προηγούμενα κεφάλαια, είναι απαραίτητα πακέτα για την εγγραφή βίντεο και κρυπτογράφηση. Αυτά δεν είναι διαθέσιμα στη βασική εγκατάσταση της Python και έτσι εγκαταστάθηκαν από εξωτερικές πηγές. Είναι το `cv2` (Computer Vision) για το χειρισμό εικόνων και βίντεο, και το `Crypto` για την κρυπτογράφηση.

Ορίζεται η στατική IP που έχει ο server (`HOST`) και η θύρα στην οποία θα αποστέλλονται τα δεδομένα (`PORT`). Αρχικοποιούνται στην τιμή `0` οι μεταβλητές `recordCount`, `encryptCount` και `sendCount`, οι οποίες χρησιμεύουν και στην ονομασία των αρχείων που προκύπτουν από το κάθε νήμα. Τα νήματα ήδη εξαναγκάζονται να τρέξουν με μια συγκεκριμένη σειρά και η ονοματοδοσία χρησιμεύει ώστε να υπάρχει μια επιπρόσθετη ασφάλεια στη σειρά εκτέλεσής τους ώστε να μην προπορεύεται το ένα από το άλλο. Όπως φαίνεται και στον κώδικα, αν για παράδειγμα το νήμα κρυπτογράφησης εκκινηθεί για κάποιο λόγο χωρίς να έχει τελειώσει το νήμα βιντεοσκόπησης (δηλαδή χωρίς να έχει αυξηθεί ο μετρητής `recordCount`), το μέρος του κώδικα για την κρυπτογράφηση δε μπορεί να εκτελεστεί. Αυτό γίνεται μόνο αν `encryptCount<recordCount` δηλαδή μόνο αν τα βιντεοσκοπημένα αρχεία είναι περισσότερα από τα κρυπτογραφημένα. Δηλαδή υπάρχουν νέα αρχεία προς κρυπτογράφηση και δεν κρυπτογραφείται κάποιο δύο φορές.

Μετά γίνεται ορισμός του κλειδιού για την αποκρυπτογράφηση. Κάποια τρίτη οντότητα που ίσως υποδυθεί τον server και λάβει τα δεδομένα δε θα μπορέσει να τα αποκρυπτογραφήσει χωρίς αυτό. Λάθος κλειδί σημαίνει ότι πρόκειται για μη εξουσιοδοτημένο άτομο και τα δεδομένα θα παραμείνουν κρυπτογραφημένα.

ΝΗΜΑ ΚΙΝΗΜΑΤΟΓΡΑΦΗΣΗΣ

Γίνεται ορισμός της μορφής των ονομάτων των αρχείων που θα κινηματογραφούνται. Το όνομα θα περιέχει την κάθε φορά τιμή της μεταβλητής `recordCount` ώστε να αλλάζει με την κάθε επανάληψη. Αυτό εισάγεται ως όρισμα σε μια άλλη μέθοδο, την `recWithCV()`.

Η `recWithCV()` ορίζει τη συσκευή καταγραφής που θα κινηματογραφήσει (το 0, που είναι το πρώτο module κάμερας από αυτά που είναι συνδεδεμένα, στην περίπτωση εδώ και το μοναδικό). Μετά ορίζονται οι διαστάσεις του βίντεο, στις μεταβλητές `videoWidth` και `videoHeight`, αντίστοιχα. Με την κλάση `VideoWriter` ορίζεται ένα προσωρινό αντικείμενο που ονομάζεται `video`, στο οποίο εισάγονται ως ορίσματα το όνομα του αρχείου βίντεο, το codec κωδικοποίησης, τα frames per second και μια λίστα που περιέχει τις διαστάσεις. Σε αυτό ακριβώς το αντικείμενο `video` αποθηκεύονται ένα ένα τα frames με τα προαναφερθέντα χαρακτηριστικά.

Τίθεται σε αυτό το σημείο ένα timestamp, ένα σημείο στο χρόνο που σηματοδοτεί την αρχή του βίντεο, στη μεταβλητή `startingTime`. Υπάρχει αναφορά σε αυτή στη συνέχεια ώστε να σταματήσει η κινηματογράφηση και να δημιουργηθεί το αρχείο.

Ξεκινάει μια επανάληψη `while True`. Από αυτή την επανάληψη εξέρχεται με `break` όταν στο επόμενο timestamp με τη μεταβλητή `endingTime`, η διαφορά τους με την `startingTime` γίνει πάνω από `10.0` (δευτερόλεπτα). Μέσα στην επανάληψη αυτή, γίνεται ανάγνωση από τη `webcam`. Από την ανάγνωση προκύπτει ένα frame, που αποθηκεύεται στη μεταβλητή `frame` και μια δυαδική τιμή που αποθηκεύεται στη `stream_ok` (ότι το καρέ αναγνώστηκε με επιτυχία). Αν η `stream_ok` είναι αληθής, προστίθεται αυτό το frame στο αντικείμενο `video`, μια ενέργεια που θα συνεχιστεί για 10 δευτερόλεπτα, όπως προαναφέρθηκε. Τελικώς, αποδεσμεύεται το ρεύμα εισόδου της κάμερας και το ρεύμα εξόδου στο `video`. Εδώ η λειτουργία της `recWithCV()` τελειώνει και επιστρέφει ο έλεγχος στη μέθοδο `record()` που αυξάνει την global μεταβλητή `recordCount` κατά 1 και τελειώνει κι αυτή, παραδίδοντας τον έλεγχο στο νήμα κρυπτογράφησης.

ΝΗΜΑ ΚΡΥΠΤΟΓΡΑΦΗΣΗΣ

Ορίζεται ότι θα γίνει επεξεργασία των global μεταβλητών `encryptCount`, `recordCount` και `key`, για να είναι δυνατή η επεξεργασία τους μέσα από το νήμα. Μετά ξεκινάει μια επανάληψη `while` με συνθήκη ελέγχου `encryptCount<recordCount` που

εξασφαλίζει ότι ο κώδικας θα εκτελεστεί μόνο όταν όντως υπάρχει αρχείο βίντεο προς κρυπτογράφηση.

Μέσα στην επανάληψη γίνεται η κατασκευή του ονόματος του αρχείου `encryptFile`, χρησιμοποιώντας την global μεταβλητή – μετρητή `encryptCount` και το άνοιγμά του. Έπειτα, πραγματοποιείται η κρυπτογράφηση με το κλειδί, με τη μέθοδο CFB του AES. Σημειώνεται ότι εισάγεται και ο Initialization Vector (`iv`) για τον πρώτο γύρο κρυπτογράφησης. Το νήμα αυξάνει την global μεταβλητή `encryptCount` κατά 1 και τελειώνει.

ΝΗΜΑ ΑΠΟΣΤΟΛΗΣ

Όπως και στα προηγούμενα νήματα, γίνεται δήλωση για τις global μεταβλητές `sendCount`, `HOST`, `PORT`, `SEPARATOR` και `BUFFER_SIZE`. Κι εδώ ξεκινάει μια επανάληψη `while` με συνθήκη ελέγχου `sendCount<encryptCount` που εξασφαλίζει ότι γίνεται προσπάθεια για αποστολή κάποιου αρχείου μόνο αν έχουν ήδη κρυπτογραφηθεί περισσότερα από όσα έχουν αποσταλεί.

Μέσα στην επανάληψη γίνεται η κατασκευή του ονόματος του αρχείου `sendFile`, χρησιμοποιώντας την global μεταβλητή – μετρητή `sendCount` και το άνοιγμά του. Στη μεταβλητή `filesize` αποθηκεύεται το μέγεθος του αρχείου. Η επόμενη ενέργεια είναι η αποστολή στον server του ονόματος ώστε να το γνωρίζει, οπότε στον server θα εμφανιστεί ακριβώς το ίδιο αρχείο με το ίδιο όνομα. Ύστερα γίνεται η αποστολή του ίδιου του αρχείου. Τελικώς αυξάνεται η global μεταβλητή `sendCount` και τερματίζεται το νήμα.

Ο έλεγχος επιστρέφει στη `main` η οποία μη έχοντας άλλο νήμα να εκτελέσει, μειώνει το μετρητή της επανάληψης κατά 1, και ξεκινάει μια νέα.

8.3 - RASPBERRY PI ZERO W – SERVER

Οι πιο περιορισμένες ικανότητες του Raspberry Pi Zero W το αντιστοιχούν στις εργασίες με μικρότερο φόρτο. Έτσι, αναμένει ως server δεδομένα σε μια συγκεκριμένη θύρα (την 50001). Η θύρα επιλέχθηκε από το εύρος των αναλώσιμων και διαθέσιμων ώστε να μην είναι κατηλειμμένη γενικά από άλλο σύνηθες πρωτόκολλο (well known ports) ούτε να απαιτείται η εφαρμογή να έχει δικαιώματα διαχειριστή για να τις χρησιμοποιήσει. Το εύρος αυτών των αναλώσιμων θυρών είναι από 49151 έως 65535.

Στη θύρα αυτή θα φτάνουν τα δεδομένα, δηλαδή με τη σειρά κάποια αρχεία. Ακόμα κι αν αυτά υποκλαπούν, ο επιτιθέμενος δε γνωρίζει ούτε τί είναι ούτε πώς να τα διαχειριστεί, πόσο μάλλον να τα αποκρυπτογραφήσει. Μπορεί να επιχειρήσει να τα κρυπταναλύσει, όμως αυτή η διαδικασία είναι γι' αυτόν χρονοβόρα, δύσκολη και τελικώς ασύμφορη αφού αν κάποια στιγμή το καταφέρει, τα αρχεία θα έχουν καταστεί πλέον μη αξιοποιήσιμα.

8.3.1 - ΚΩΔΙΚΑΣ SERVER ΚΑΙ ΕΠΕΞΗΓΗΣΗ

```
from Crypto.Cipher import AES
from Crypto.Util.Padding import pad, unpad
from base64 import b64encode, b64decode

import sys
import socket
import subprocess
import os
import threading

decryptCount = 0
cleanCount = 0
key = "thisK3yW1llN3v3rB3F0und@@"
key = key.encode('UTF-8')
```

```
key = pad(key, AES.block_size)

# Network things
BUFFER_SIZE = 4096
SEPARATOR = "<SEPARATOR>"

SERVER_HOST = "0.0.0.0"
SERVER_PORT = 50001
# 0.0.0.0 for all IPv4 addresses on the local machine
# That means the server can be accessed whatever IP it has
# SERVER_PORT is set from the expendable range



# ================================================
#    file_catch          File reception method
# ================================================
def file_catch(BUFFER_SIZE, SEPARATOR):
        # Creating server socket object
        s = socket.socket()
        # Bind the socket to the above host and port
        s.bind((SERVER_HOST, SERVER_PORT))

        # Listening for connections, max 1
        s.listen(1)
        print(f"[+] Listening as {SERVER_HOST}:{SERVER_PORT}")
        # Accepting connections ...
        client_socket, address = s.accept()

        print(f"[+] {address} is connected.")
        print("Server starting...")
        # Here the server is ready to receive the file

        # This receives file information
        received = client_socket.recv(BUFFER_SIZE).decode()
        filename, filesize = received.split(SEPARATOR)
        # Removes absolute path, if any
        filename = os.path.basename(filename)
        filesize = int(filesize)

        with open(filename, "wb") as f:
                while True:
                        bytes_read = client_socket.recv(BUFFER_SIZE)
                        if not bytes_read:
                                break
                        f.write(bytes_read)
        client_socket.close()
        s.close()


# ================================================
#    decrypt                Decryption method
# ================================================
def decrypt(filename, productCounter, key):
        with open(filename, 'r') as entry:
                try:
                        data = entry.read()
                        length = len(data)

                        # iv is the initialization vector
                        iv = data[:24]
                        iv = b64decode(iv)
                        ciphertext = data[24:length]
                        ciphertext = b64decode(ciphertext)
                        cipher = AES.new(key, AES.MODE_CFB, iv)
```

```
                        decrypted = cipher.decrypt(ciphertext)
                        decrypted = unpad(decrypted, AES.block_size)

                        #  Write all this to a new file
                        cleanFile = "clean" + str(productCounter) + ".avi"
                        with open(cleanFile, 'wb') as data:
                                data.write(decrypted)
                        data.close()
                except(ValueError, KeyError):
                        print('Wrong password.')



# ================================================
#    file_exists         Check file existence method
# ================================================
def file_exists(counter):
        file_to_check = "encrypted" + str(counter) + ".enc"
        if os.path.exists(file_to_check):
                return True
        else:
                return False


# ================================================================================
#    reception_function      A receiving function to constantly receive files
# ================================================================================
def reception_function():
        global BUFFER_SIZE
        global SEPARATOR
        global decryptCount
        while True:
                print("Waiting to receive...")
                file_catch(BUFFER_SIZE, SEPARATOR)
                decryptCount = decryptCount + 1
                print("A file was received successfully!")


# ================================================================================
#    decryption_function     A function to decrypt files, if it finds them
# ================================================================================
def decryption_function():
        global decryptCount
        global cleanCount
        global key
        print("Waiting to decrypt...")
        while file_exists(decryptCount):
                file_to_decrypt = "encrypted" + str(decryptCount) + ".enc"
                decrypt(file_to_decrypt, cleanCount, key)
                decryptCount = decryptCount + 1
                cleanCount = cleanCount + 1
                print("A file has been decrypted successfully!")


# ===================================== #
#   A C T U A L   S E R V E R   C O D E   #
# ===================================== #
if __name__ == "__main__":
        print("main function is running...")

        while True:
                # Reception thread
                reception_thread_instance =
threading.Thread(target=reception_function)
```

```
                    # Decryption thread
                    decryption_thread_instance =
threading.Thread(target=decryption_function)

                    # Thread initialization
                    reception_thread_instance.start()
                    reception_thread_instance.join()

                    decryption_thread_instance.start()
                    decryption_thread_instance.join()
```

Εδώ τα νήματα είναι δύο, καθώς δε χρειάζεται κινηματογράφηση. Η βασική μέθοδος του καθενός καλεί υπομεθόδους που εκτελούν επιμέρους εργασίες και βοηθούν καλύτερα την κατανόηση του κώδικα. Οι βασικές μέθοδοι των νημάτων παύουν την κανονική ροή του προγράμματος όπως και στον client.

ΑΡΧΙΚΕΣ ΣΥΜΒΑΣΕΙΣ

Στην αρχή γίνονται τα απαραίτητα import. Μετά, για να επιτευχθούν οι αντίστοιχοι περιορισμοί με αυτούς στον client χρειάζονται δύο μεταβλητές, η `decryptCount` που μετράει τα κρυπτογραφημένα αρχεία και η `cleanCount` για τα αποκρυπτογραφημένα. Το κλειδί είναι προφανώς όμοιο με του client και μετά τον ορισμό του γίνεται αποκωδικοποίηση με UTF-8 και συμπληρώνεται (pad) με χαρακτήρες στο τέλος για να έχει το μέγεθος που απαιτεί ο AES.

Ύστερα ορίζεται το μέγεθος του `BUFFER_SIZE` ώστε το ληφθέν αρχείο να έρχεται σε κομμάτια. Ο διαχωριστής (`SEPARATOR`) είναι μια γενική μεταβλητή που χωρίζει το όνομα αρχείου από την κατάληξή του.

Η πόρτα (port), όπως έχει προαναφερθεί, είναι η `50001` από το εύρος των αναλώσιμων. Στη μεταβλητή HOST τίθεται `0.0.0.0` και όχι `192.168.1.30` που είναι η IP του server. Αυτό θα επιτρέψει στον server να δέχεται δεδομένα ό,τι IP και να έχει, ως μέτρο ασφάλειας στη λήψη των δεδομένων.

ΝΗΜΑ ΛΗΨΗΣ

Η βασική μέθοδός του, η `reception_function` καλεί την υπομέθοδο `file_catch`. Η τελευταία δημιουργεί το `socket` και έτσι ο server αναμένει. Μετά, λαμβάνει πληροφορίες σχετικά με το αρχείο μέσω του `socket`, και το διαχωρίζει σε όνομα και κατάληξη, αφαιρώντας τυχόν απόλυτο μονοπάτι.

Στην επανάληψη while λαμβάνει τα δεδομένα από τον client ανά κομμάτια μεγέθους `BUFFER_SIZE`. Αν δεν υπάρχουν άλλα δεδομένα προς ανάγνωση από τον client (`if not bytes_read`) γράφονται τα υπολειπόμενα στο αντικείμενο `f` (το αρχείο) και κλείνει το `socket`.

Επιστρέφοντας στην βασική μέθοδο του νήματος `reception_function`, αυξάνει ο μετρητής `decryptCount` που μετράει τα κρυπτογραφημένα αρχεία που λαμβάνονται και το νήμα τελειώνει.

ΝΗΜΑ ΑΠΟΚΡΥΠΤΟΓΡΑΦΗΣΗΣ

Η βασική μέθοδος εδώ, η `decryption_function`, καλεί δύο υπομεθόδους. Πρώτα την `file_exists`, που βάσει του ονόματος αρχείου που αναμένεται, ελέγχει αν όντως υπάρχει, δηλαδή αν έχει ληφθεί στον τρέχοντα φάκελο. Το όνομα του αρχείου που αναμένεται είναι γνωστό οπότε είναι δυνατός αυτός ο έλεγχος. Επιστρέφει `True` ή `False` αναλόγως.

Αυτή η δυαδική τιμή ελέγχεται από μια `while` στη βασική μέθοδο, συνεχώς, μέχρι το αρχείο να εμφανιστεί και η εκτέλεση δεν προχωράει, σε αντίθετη περίπτωση. Μέσα στη `while` καλείται η `decrypt` που κάνει την αποκρυπτογράφηση.

Η `decrypt` διαβάζει από το `filename` το κρυπτογραφημένο αρχείο. Αφού χρησιμοποιείται ο τρόπος λειτουργίας CFB του AES, υπάρχει ο Initialization Vector και είναι οι πρώτοι χαρακτήρες, οπότε διαχωρίζονται από τα υπόλοιπα κρυπτογραφημένα δεδομένα, με το κλειδί που ήδη υπάρχει. Η αποκρυπτογράφηση γίνεται με την ίδια μέθοδο (CFB) και γράφονται

τα δεδομένα σε ένα νέο αρχείο.

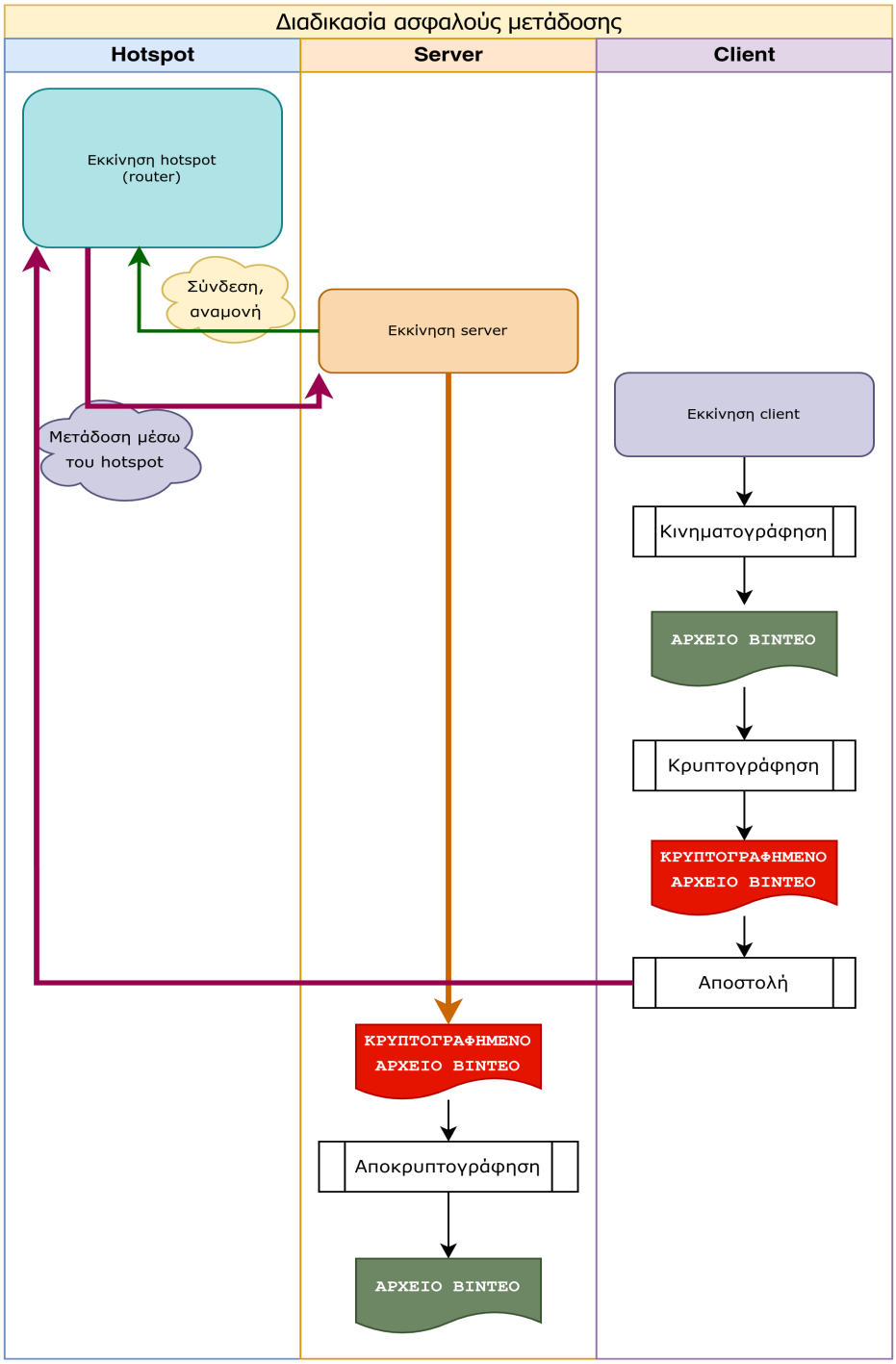
Στη βασική μέθοδο αυξάνει ο μετρητής `cleanCount` των αποκρυπτογραφημένων αρχείων και και το νήμα ολοκληρώνει την εργασία του.

Στη main δεν ορίζονται δύο επαναλήψεις για τη λήψη των δύο αρχείων του client, αλλά ο server αφήνεται μόνιμα ελεύθερος να λαμβάνει δεδομένα. Απλώς δημιουργούνται δύο αντικείμενα νημάτων και μετά, όπως και στον client, ξεκινάει κάθε νήμα σταματώντας τη ροή του προγράμματος μέχρι να τελειώσει την εργασία του πριν συνεχίσει στην έναρξη του επόμενου. Αν ορίσουμε τον client να αποστείλει περισσότερα αρχεία ή και να στέλνει μόνιμα, δε χρειάζεται καμία τροποποίηση στον κώδικα του server.

ΚΕΦΑΛΑΙΟ 9
ΑΝΑΛΥΣΗ ΔΟΚΙΜΗΣ

9.1 – ΠΕΡΙΓΡΑΦΗ ΔΟΚΙΜΗΣ

Σε αυτό το κεφάλαιο θα πραγματοποιηθεί δοκιμή της υλοποίησης που περιγράφηκε θεωρητικά ως τώρα. Αρχικά ενεργοποιείται το router (hotspot) κι έπειτα ο server ώστε να λάβει static IP από το hotspot και να τρέξει το πρόγραμμά του ώστε να αναμένει δεδομένα. Έπειτα ενεργοποιείται ο client, και ξεκινάει το πρόγραμμά του ‘ μέσω socket συνδέεται κι αυτός στο hotspot και αποστέλλει αυτοματοποιημένα τα δεδομένα.



Εικόνα 9.1
Απεικόνιση διαδικασίας ασφαλούς μετάδοσης αρχείου

Στην παραπάνω εικόνα παρουσιάζεται η διαδικασία με την οποία μεταδίδεται με ασφάλεια ένα αρχείο μεταξύ client και server. Όπως διαφαίνεται, το μόνο που αποστέλλεται over the air μέσω του hotspot είναι κρυπτογραφημένα δεδομένα.

Στον προκαθορισμένο φάκελο αρχείων στον server κάθε φορά εμφανίζεται ένα αρχείο το οποίο είναι το απεσταλμένο κρυπτογραφημένο βίντεο και σχεδόν αμέσως μετά εμφανίζεται κι άλλο, το οποίο είναι η αποκρυπτογράφησή του από τον ίδιο τον server, ένα αρχείο βίντεο στο οποίο φαίνεται ό,τι κινηματογράφησε ο client. Αυτό θα συμβεί άλλες δύο φορές αφού στον client έχουν οριστεί δύο επαναλήψεις.

9.2 – ΕΚΚΙΝΗΣΗ ΔΟΚΙΜΗΣ

Αρχικά συνδέεται το router με τροφοδοσία ρεύματος. Μετά από κάποιο χρόνο το ενσωματωμένο σύστημά του θα ξεκινήσει και θα κάνει broadcast την SSID του. Ο κωδικός πρόσβασης στο Wifi και οι λοιπές ρυθμίσεις είναι ήδη προαποθηκευμένες οπότε δεν απαιτείται καμία άλλη ενέργεια.

Έπειτα, συνδέεται και το Raspberry Pi Zero με τροφοδοσία ρεύματος. Το λειτουργικό σύστημα Raspberry Pi OS ξεκινάει, διαπιστώνει ότι υπάρχει SSID εντός εμβέλειας της οποίας τον κωδικό του WiFi γνωρίζει, και συνδέεται στο router αυτόματα μιας και έχει ήδη συνδεθεί ξανά. Δεν υπάρχει περίπτωση να συνδεθεί κατά λάθος σε άλλο router αφού είναι και το μοναδικό στο οποίο το Raspberry Pi Zero έχει συνδεθεί ποτέ. Το router, βλέποντας την MAC διεύθυνση της κάρτας δικτύου του, θα του αποδώσει τη static IP 192.168.0.31 όπως έχει ρυθμιστεί.

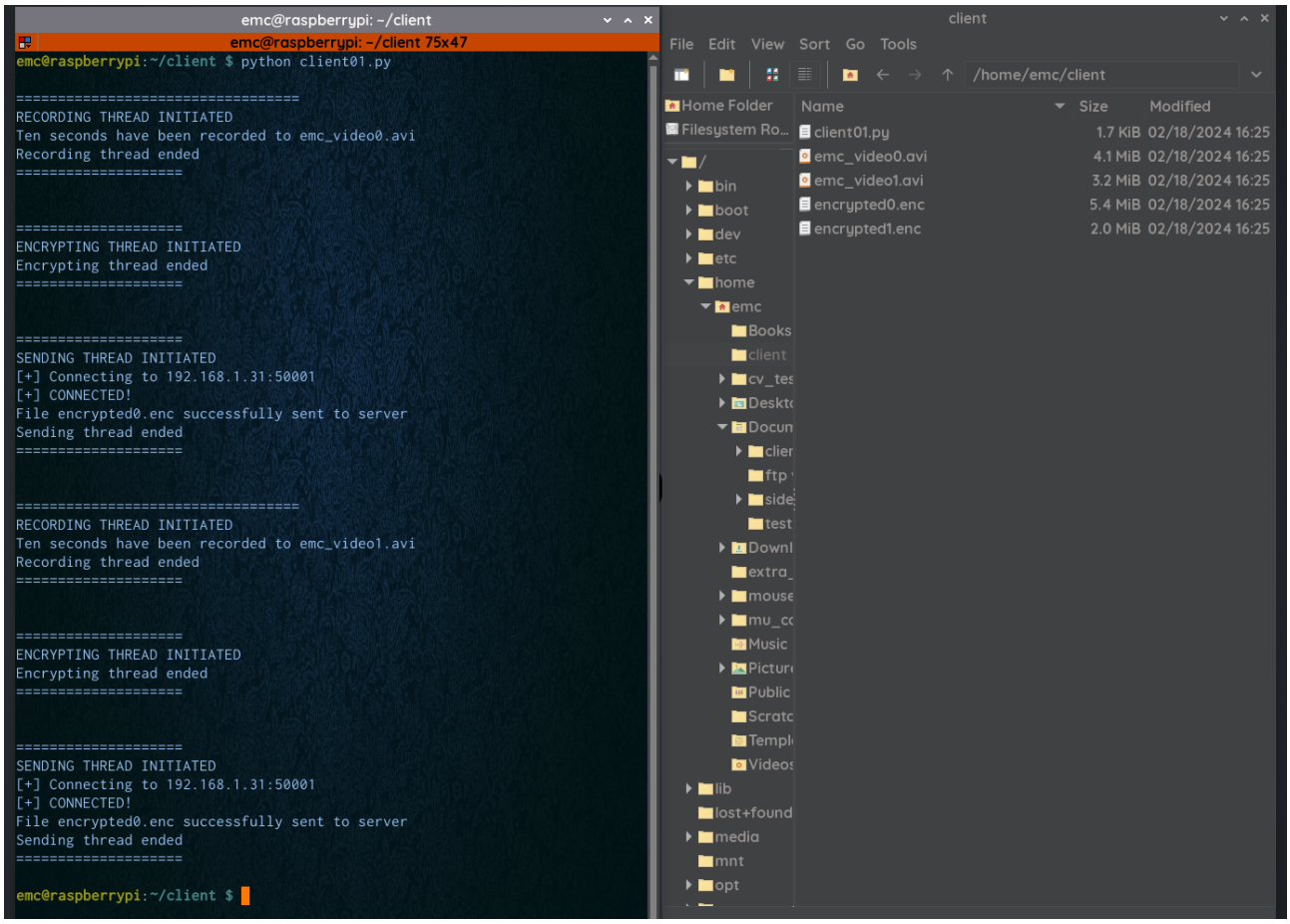
Μετά από αυτό στον server εκκινείται ένα τερματικό στο φάκελο που υπάρχει το Python αρχείο του server, τον `/home/emc/server` και τρέχει με την εντολή:

```
$ python server01.py
```

Τώρα ο server τρέχει. Η IP του είναι γνωστή, αυτή που ορίστηκε στατικά στο Raspberry Pi Zero. Επίσης η πόρτα (port) που αναμένει είναι γνωστή (η 50001) και ορισμένη στον κώδικα. Σε αυτό το σημείο πρέπει να συνδεθεί και το Raspberry Pi 4 στην τροφοδοσία, το οποίο με τη σειρά του θα εκκινήσει το λειτουργικό του σύστημα και με όμοιο τρόπο θα συνδεθεί στο ίδιο router, λαβαίνοντας την IP 192.168.0.30. Εκκινείται τερματικό και ομοίως δίνεται μια αντίστοιχη εντολή για να τρέξει το αντίστοιχο Python αρχείο του client.

```
$ python client.py
```

Ο client ξεκινάει την εργασία του. Σε καίρια σημεία έχουν τοποθετηθεί διάφορα `print` ώστε να ενημερώνουν για τις ενέργειες που εκτελούνται. Όπως για παράδειγμα στην έναρξη της διαδικασίας, στη σύνδεση με τον server, στην έναρξη και ενέργεια του κάθε thread. Στο παρακάτω screenshot φαίνεται αριστερά το τρέξιμο και output του client και δεξιά το περιεχόμενο του αντίστοιχου φακέλου. Εκεί έχουν εμφανιστεί δύο αρχεία βίντεο, με κατάληξη "`.avi`", και δύο κρυπτογραφημένα αρχεία, με κατάληξη "`.enc`". Τα τελευταία είναι που αποστέλλονται στον server.

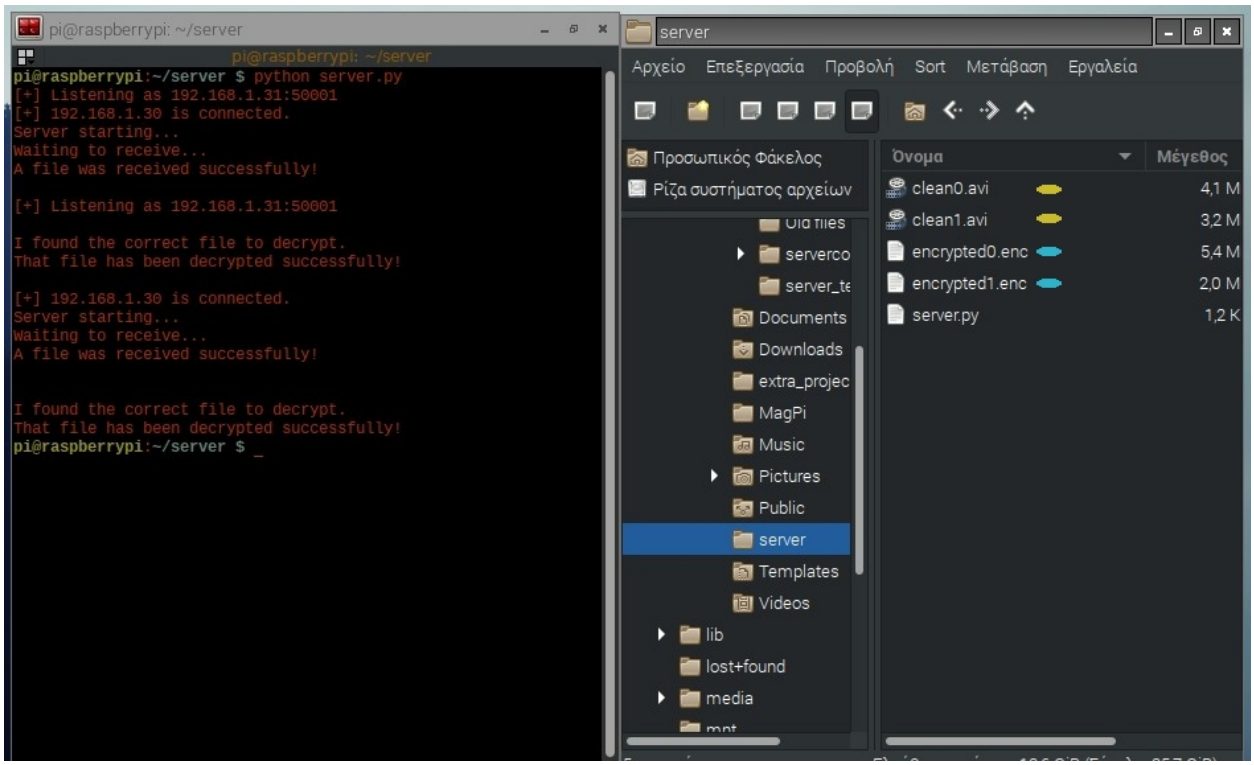


Εικόνα 9.2

Ο client μετά την αποστολή των κρυπτογραφημένων αρχείων

Με την επιστροφή στον server που ήταν σε κατάσταση αναμονής δεδομένων παρατηρείται η σταδιακή εμφάνιση μιας σειράς αρχείων. Πρώτα εμφανίζεται ένα αρχείο με κατάληξη ".enc", η οποία υποδεικνύει τυπικά ένα κρυπτογραφημένο αρχείο. Είναι δυνατός ο ορισμός οποιασδήποτε άλλης κατάληξης ή και καθόλου. Η κατάληξη χρησιμεύει καθαρά ενημερωτικά ώστε στην προκειμένη περίπτωση να φαίνεται το ζητούμενο καλύτερα.

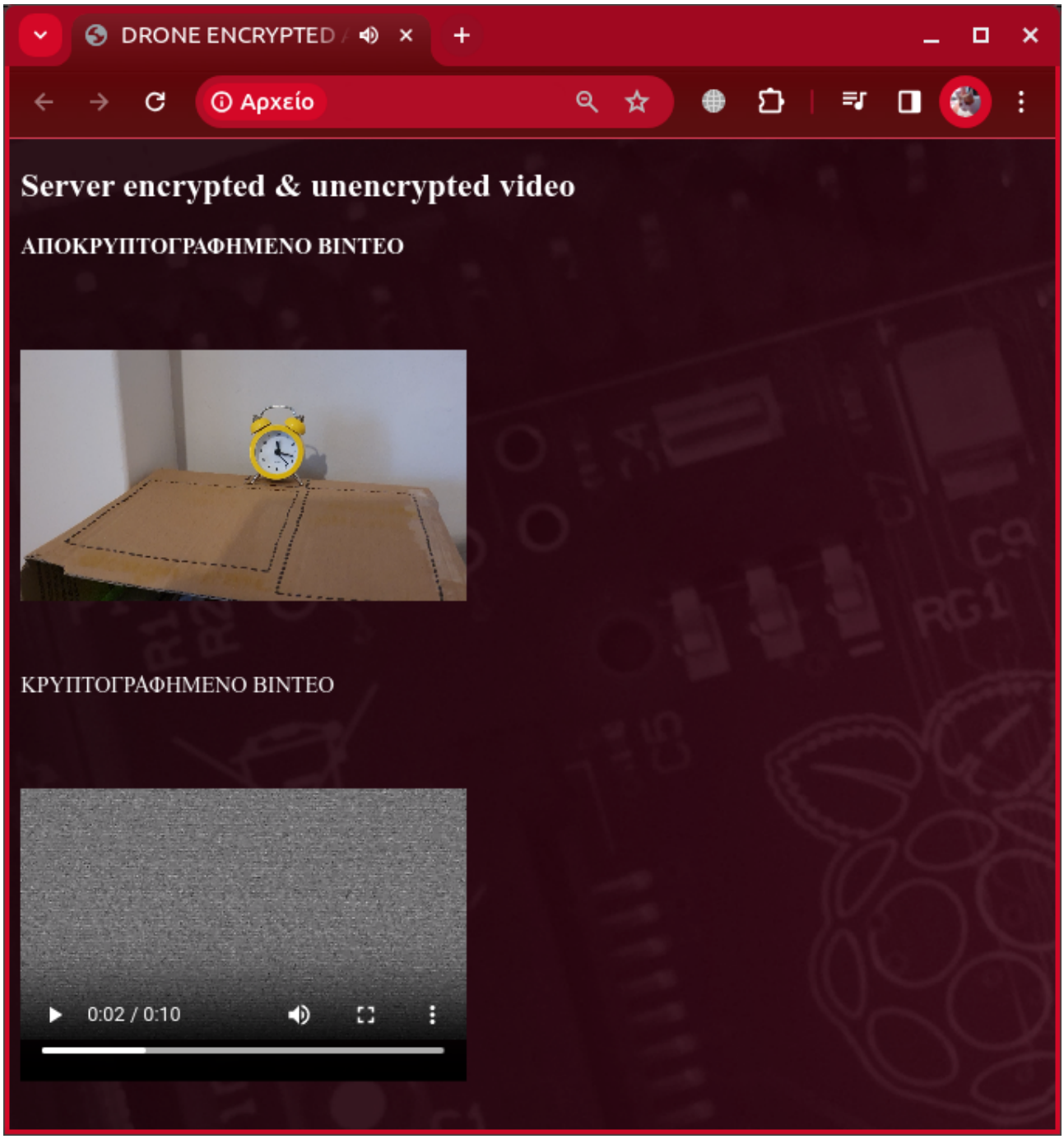
Αυτό είναι το αρχείο όπως παρελήφθη από τον client. Στο επόμενο screenshot φαίνεται αριστερά το τρέξιμο και output του server και δεξιά το περιεχόμενο του αντίστοιχου φακέλου.



Εικόνα 9.3
Περιεχόμενο προεπιλεγμένου φακέλου χρήστη στον client

Αυτό που συμβαίνει στον server είναι ότι ξεκινάει την αποκρυπτογράφηση του αρχείου όσο ο client ταυτοχρόνως κινηματογραφεί, κρυπτογραφεί και αποστέλλει το επόμενο. Μετά από λίγο εμφανίζεται στον ίδιο φάκελο το αρχείο `clean0.avi` που είναι το αποτέλεσμα της αποκρυπτογράφησης από τον server. Ο server είναι έτοιμος να λάβει το επόμενο αρχείο, το οποίο είναι και το τελευταίο αφού έχουν οριστεί δύο επαναλήψεις. Μετά το τέλος της εργασίας του client, υπάρχουν στον server τελικώς δύο κρυπτογραφημένα (γαλάζιο χρώμα) και δύο αποκρυπτογραφημένα (κίτρινο χρώμα) αρχεία.

Έχει κατασκευαστεί μια τυπική σελίδα html όπου παρατίθενται τα αρχεία στο ίδιο πλάνο ώστε να γίνει η σύγκριση. Στην είκονα φαίνεται πάνω το πρώτο από τα αποκρυπτογραφημένα αρχεία και κάτω το αντίστοιχο κρυπτογραφημένο του με κουμπιά ελέγχου και μπάρα κύλισης στο καθένα. Οπότε κάποιο άτομο που τυχόν υποκλέψει κάποιο αρχείο και επιχειρήσει να το αναπαράγει, θα δει “χιόνια”.



Εικόνα 9.4
Αποκρυπτογραφημένο και κρυπτογραφημένο αρχείο σε παράθεση

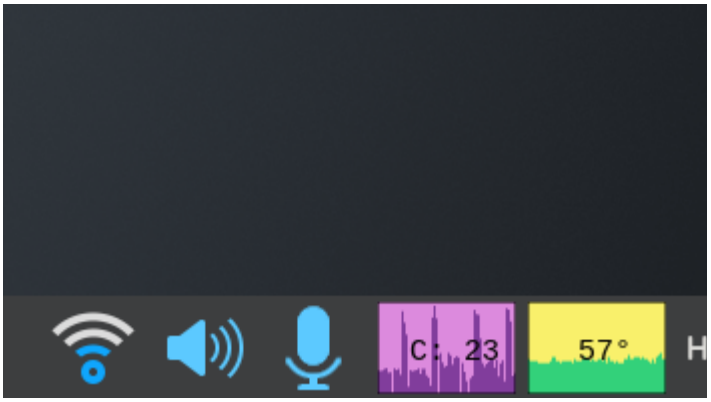
9.3 – ΘΕΡΜΟΚΡΑΣΙΑ ΔΟΚΙΜΗΣ

Θα παρατηρηθεί σε όλη την παραπάνω διαδικασία την αυξομείωση θερμοκρασίας στον client που, αφενός μεν εκτελεί την πιο βαριά εργασία αφετέρου δε, λειτουργεί με μπαταρία. Γίνεται μνεία ότι δε χρησιμοποιήθηκαν ούτε heat sink ούτε ανεμιστήρας, ώστε να παρατηρηθεί καλύτερα το φαινόμενο thermal throttling.

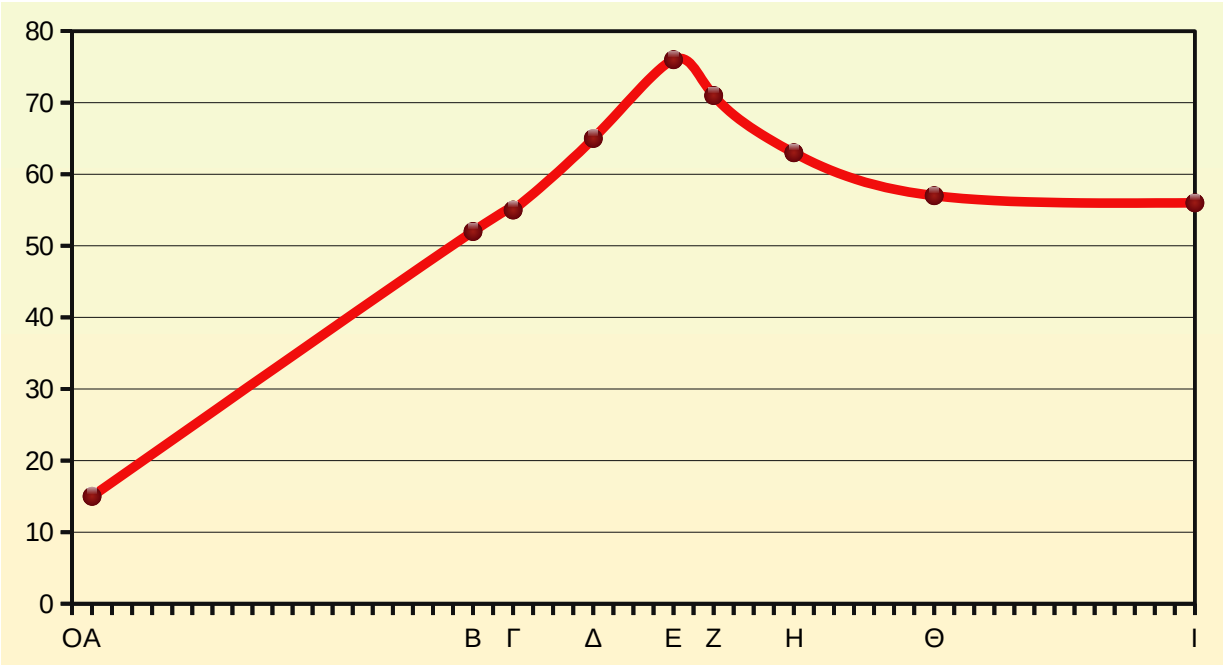
Οι μετρήσεις έγιναν σε κομβικά σημεία της διαδικασίας έχοντας θέσει την επανάληψη σε ατέρμονα βρόχο. Δηλαδή στον κώδικα του client που στην επανάληψη των νημάτων είχε οριστεί ο μετρητής `counter=2` για την αποστολή δύο αρχείων, τώρα για να παρατηρηθεί η θερμοκρασία έχει τεθεί `while True`. Οι υποδιαιρέσεις στον άξονα των x είναι ανά 10

δευτερόλεπτα.

Μεταβαίνουμε στις Ρυθμίσεις του Ταμπλό, όπου έχουν προστεθεί εφαρμογές που δείχνουν και τη θερμοκρασία και τη χρήση CPU (σε %) για άμεσες παρατηρήσεις σε πραγματικό χρόνο.



Εικόνα 9.5
Εφαρμογές Ταμπλό για τη χρήση επεξεργαστή και τη θερμοκρασία του



Εικόνα 9.6
Μετρήσεις θερμοκρασίας κατά τη δοκιμή

Σημείο Ο: Εδώ το Raspberry Pi είναι σβηστό. Ο επεξεργαστής είναι σε θερμοκρασία δωματίου.
Σημείο Α: Με την εκκίνηση του συστήματος η θερμοκρασία αυξάνεται. Η πρώτη ένδειξη που παρατηρείται με την έναρξη του γραφικού περιβάλλοντος είναι 15°C.
Σημείο Β: Αφήνεται το σύστημα να λειτουργήσει 3 λεπτά ώστε η θερμοκρασία να φτάσει στο μέγιστο σημείο της και να σταθεροποιηθεί. Παρατηρείται η ένδειξη 52°C.
Σημείο Γ: Εδώ ξεκινάει το πρόγραμμα του client και παρατηρείται μετά από 10 δευτερόλεπτα η θερμοκρασία στους 55°C.
Σημείο Δ: 30 δευτερόλεπτα αργότερα παρατηρείται η θερμοκρασία στους 65°C.
Σημείο Ε: Άλλα 30 δευτερόλεπτα μετά παρατηρείται η θερμοκρασία να έχει αυξηθεί στους 76°C.
Σημείο Ζ: Άλλα 10 δευτερόλεπτα μετά παρατηρείται η θερμοκρασία να αρχίζει να μειώνεται, πέφτοντας στους 71°C. Εδώ το thermal throttling φαίνεται ότι ξεκινήσει.
Σημείο Η: Επειδή η θερμοκρασία μειώνεται με πιο αργούς ρυθμούς, η επόμενη παρατήρηση γίνεται μετά από 30 δευτερόλεπτα και είναι στους 63°C.

Σημείο Θ: Το επόμενο σημείο είναι μετά από 60 δευτερόλεπτα το οποίο πλησιάζει τη θερμοκρασία ασφαλείας, στους 57°C.
Σημείο Ι: Η επόμενη μέτρηση είναι μετά από 120 δευτερόλεπτα. Η θερμοκρασία έχει πέσει στους 56°C και ουσιαστικά από εκεί και έπειτα παραμένει σταθερή.

Παρατηρείται ότι η θερμοκρασία του επεξεργαστή, με την εκκίνηση της εργασίας του προγράμματος, αυξάνεται εύκολα και γρήγορα. Σύντομα όμως τίθεται σε λειτουργία το thermal throttling που σταδιακά την επαναφέρει σε αποδεκτά επίπεδα. Αξίζει να σημειωθεί πως, ακόμα και με τη μειωμένη απόδοση της CPU, δε διακόπηκε ούτε εμποδίστηκε η εκτέλεση του προγράμματος, που εκτελέστηκε απρόσκοπτα.

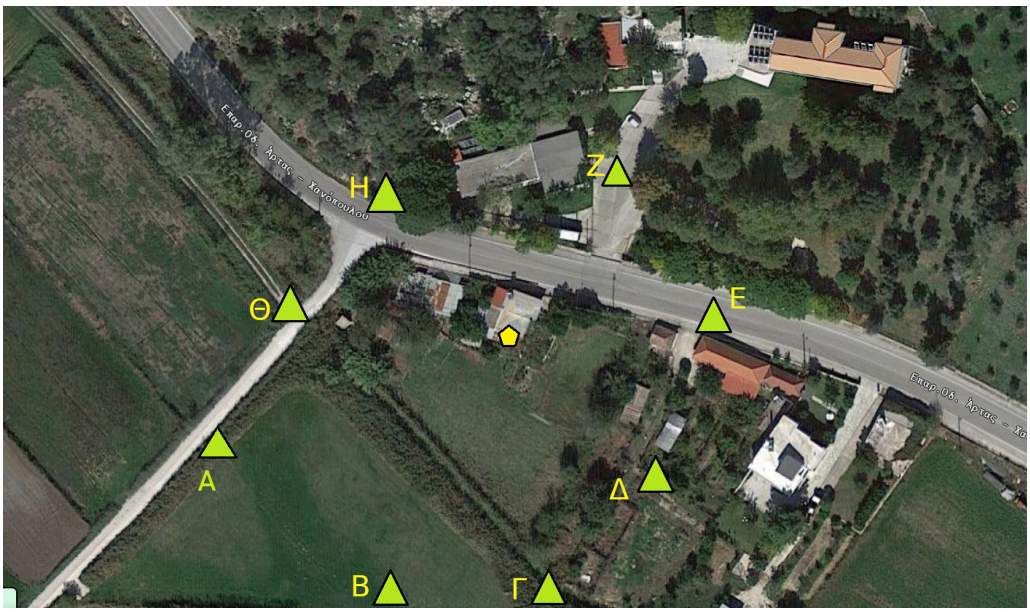
Όσο για τη διάρκεια λειτουργίας του συστήματος drone – Raspberry Pi, πρέπει να αναφερθεί ότι τα drone του εμπορίου για ιδιώτες έχουν διάρκεια πτήσης περίπου 40 λεπτά. Το Raspberry Pi λειτουργεί με την προαναφερθείσα power bank των 5000 mAh για 10 με 12 ώρες σε αναμονή, και περίπου 5 με 6 ώρες εκτελώντας εργασίες, η οποία διάρκεια υπερκαλύπτει σε μεγάλο βαθμό τη διάρκεια πτήσης ώστε η ασφαλής μετάδοση να πραγματοποιείται αδιάλειπτα.

9.4 – ΕΜΒΕΛΕΙΑ ΣΥΣΤΗΜΑΤΟΣ

Σε αυτό το τμήμα θα παρουσιαστεί μια εικόνα της απόστασης που μπορεί να έχει ο client από το router (hotspot). Επειδή το σύστημα προορίζεται για την ενσωμάτωση σε drone, οι μετρήσεις γίνονται σε ανοιχτό χώρο.

Έχει επιλεγεί μια αγροτική περιοχή και, με το router σταθερό απομακρύνεται ο client από αυτό. Παρατηρώντας συνεχώς μέσω των Ρυθμίσεων του router τις συνδεδεμένες σε αυτό συσκευές, όταν ο client αποσυνδεθεί θα σημειωθεί η απόσταση μέχρι εκεί. Σημειώνοντας αρκετά σημεία, με και χωρίς εμπόδια ανάμεσά τους, θα γίνει φανερή μια ολοκληρωμένη εικόνα της εμβέλειας λειτουργίας. Η εμβέλεια είναι συνολική προς όλες τις κατεθύνσεις και το σχήμα της μοιάζει με ένα ημισφαιρικό θόλο με ακτίνα αυτή την εμβέλεια, στο κέντρο του οποίου είναι το router. Ο θόλος μειώνει την ακτίνα του αν υπάρχουν εμπόδια ανάμεσα σε client και router.

Εκκινείται η διαδικασία όπως και παραπάνω για τη μέτρηση θερμότητας, με `while True` στην επανάληψη. Η απώλεια σύνδεσης μπορεί να διαπιστωθεί με δύο τρόπους, είτε να σημειωθούν σε ποιές αποστάσεις ο server δε λαμβάνει δεδομένα, που σημαίνει ότι ο client έχει απομακρυνθεί τόσο πολύ που έχει βγει από την εμβέλεια του WiFi είτε απλώς μέσω των Ρυθμίσεων του router, όταν το Raspberry Pi 4 χαθεί από τη λίστα συνδεδεμένων συσκευών.



Εικόνα 9.7
Μετρήσεις αποστάσεων

Σημείο Α: Ανοιχτός χώρος χωρίς εμπόδια (84 m).
Σημείο Β: Ανοιχτός χώρος χωρίς εμπόδια (79 m).
Σημείο Γ: Ανοιχτός χώρος με λίγα δέντρα στο οπτικό πεδίο (70 m).
Σημείο Δ: Λίγα δέντρα και οικήματα στο οπτικό πεδίο (48 m).
Σημείο Ε: Βλάστηση και αρκετά οικήματα στο οπτικό πεδίο (46 m).
Σημείο Ζ: Ανάμεσα σε router και Raspberry Pi υπάρχουν τοίχοι από πολλά κτίρια, λίγη βλάστηση και ένα λεωφορείο (47 m).
Σημείο Η: Ανάμεσα σε router και Raspberry Pi υπάρχουν τοίχοι από πολλά κτίρια, και πυκνή βλάστηση (46 m).
Σημείο Θ: Ανάμεσα σε router και Raspberry Pi υπάρχει πυκνή βλάστηση (60 m).

Το σημείο που είναι τοποθετημένο το router είναι το κίτρινο πεντάγωνο στο κέντρο. Φαίνεται ότι η εμβέλεια του WiFi επηρεάζεται με την ύπαρξη εμποδίων. Μειώνεται όταν υπάρχουν μικρά εμπόδια όπως δέντρα και φυτά ενώ πέφτει δραματικά όταν υπάρχουν κτίρια και τοίχοι ανάμεσα στα σημεία.

Σημειώνεται ότι η εμβέλεια του drone πάνω στο οποίο το Raspberry Pi κάθε φορά θα ενσωματώνεται μπορεί να διαφέρει από αυτή του WiFi του Raspberry Pi και του router. Αν του drone είναι μεγαλύτερη, αυτό σημαίνει ότι από ένα σημείο και έπειτα, ενώ το drone συνεχίζει να πετάει, ο σταθμός εδάφους θα σταματήσει να λαμβάνει δεδομένα, αφού το Raspberry Pi θα έχει αποσυνδεθεί.

ΚΕΦΑΛΑΙΟ 10
ΤΕΛΙΚΑ ΣΥΜΠΕΡΑΣΜΑΤΑ

Από την υλοποίηση του όλου εγχειρήματος που παρουσιάστηκε παραπάνω, συμπεραίνονται τα εξής.

- Οι Single Board Computers είναι μια οικονομική λύση που μπορεί όχι μόνο να επιτελέσει βαριές εργασίες αλλά και να εκκινήσει αυτόνομα ολοκληρωμένη έκδοση λειτουργικού συστήματος. Η ασφαλής μετάδοση δεδομένων μεταξύ Raspberry Pi έγινε με το γραφικό περιβάλλον σε λειτουργία. Μπορεί όμως το ίδιο να επιτευχθεί και *headless* δηλαδή να οριστούν οι client και server να ξεκινήσουν με γραμμή εντολών (τερματικό) και απλώς να τρέξουν αυτοματοποιημένα τα αρχεία Python, ο καθένας.
- Το Raspberry Pi είναι ένα δυνατό μηχάνημα, εξοπλισμένο με πληθώρα από interfaces (USB, serial, wireless, GPIO) και ικανό να χρησιμοποιηθεί σε πολλούς τομείς. Εδώ χρησιμοποιήθηκε κάμερα USB όμως υπάρχουν και πιο μικρά modules κάμερας που συνδέονται στη σειριακή θύρα, με διάφορες αναλύσεις και δυνατότητες (πχ υπέρυθρες, θερμικές, fisheye).
- Μια τυπική κρυπτογράφηση μπορεί να επιτυγχάνεται σχετικά απλά, με τη χρήση βιβλιοθηκών και πακέτων λογισμικού. Σε μεγαλύτερη κλίμακα όμως είναι όμως ένας συνεχής αγώνας μαθηματικών ανάμεσα στους κρυπτογράφους και τους κρυπταναλυτές για μια νέα επαναστατική μέθοδο που θα απλοποιήσει και θα δώσει ώθηση στους σκοπούς τους, καθώς διακυβεύονται δεδομένα μεγαλύτερου όγκου και σοβαρότητας.
- Η γλώσσα προγραμματισμού Python καθιστά απλή τη συγγραφή πολύπλοκων προγραμμάτων με λίγες, απλές και άμεσα κατανοητές γραμμές κώδικα. Στο δικό μας πείραμα χρησιμοποιήσαμε κάποια εξωτερικά πακέτα για όραση υπολογιστών και επικοινωνία μέσω δικτύου. Η Python έχει τη δυνατότητα να ενσωματώσει στο ίδιο πρόγραμμα γραμμές κώδικα που έχουν συνταχθεί σε άλλες γλώσσες προγραμματισμού και να χρησιμοποιηθούν αυτούσιες χωρίς καμία αλλαγή.
- Η εμβέλεια του WiFi μειώνεται όταν υπάρχουν μικρά εμπόδια όπως δέντρα και φυτά ενώ πέφτει δραματικά όταν υπάρχουν κτίρια και τοίχοι ανάμεσα στα σημεία. Η εμβέλεια του drone πάνω στο οποίο το Raspberry Pi κάθε φορά ενσωματώνεται μπορεί να διαφέρει από αυτή του WiFi του Raspberry Pi και του router. Αν του εκάστοτε drone είναι μεγαλύτερη, από ένα σημείο πτήσης και έπειτα, ο σταθμός εδάφους θα σταματήσει να λαμβάνει δεδομένα, αφού το Raspberry Pi θα έχει αποσυνδεθεί.
- Η ενσωματωμένη ικανότητα του thermal throttling λειτούργησε όταν έπρεπε ώστε η θερμοκρασία του επεξεργαστή να πέσει σε ασφαλή για το υλικό του επίπεδα, σύμφωνα με τον κατασκευαστή.
- Τυχόν υποκλαπέντα δεδομένα φαίνεται εξαρχής ότι είναι παντελώς μη αξιοποιήσιμα. Ο υποκλέπτων δε γνωρίζει τι είδους αρχεία μεταδίδονται. Ακόμα κι αν το μαντέψει, έρχεται αντιμέτωπος με την ισχύ του αλγορίθμου AES, για την κρυπτανάλυση του οποίου θα χρειαστεί τόσο χρόνο ώστε η εκμετάλλευση των δεδομένων θα είναι μηδενική.

Η χρήση drone έχει αυξηθεί ραγδαία τα τελευταία χρόνια λόγω της βελτίωσης της τεχνολογίας γενικότερα και την πτώση της τιμής κατασκευής τους που τα έθεσε προσιτά στο ευρύ κοινό, σε μεγάλο εύρος ποιοτήτων, τιμών, μοντέλων και τύπων. Η στιβαρότερη κατασκευή τους και η αυξημένη εμβέλειά τους έκανε απαραίτητη την προσθήκη κάμερας ώστε ο χειριστής να βλέπει την κίνησή του. Σε περιπτώσεις, είναι δυνατόν να χρειάζονται περισσότεροι χειριστές, ένας για την πλοήγηση του drone, ένας για τον έλεγχο της κάμερας και την κινηματογράφηση και άλλοι για τα τυχόν πρόσθετα εργαλεία ή αντικείμενα που μεταφέρονται.

Η κινηματογράφηση βίντεο με drone εφαρμόζεται ήδη σε πολλούς τομείς όπως ιδιωτικά ως live streaming σε διαδικτυακές πλατφόρμες και στη βιομηχανία του θεάματος. Χρησιμοποιείται επίσης στη συλλογή πληροφοριών μετά από καταστροφές και την αποστολή βοήθειας. Επίσης θερμικές κάμερες που εντοπίζουν άτομα ή χαμένα αντικείμενα

χρησιμοποιούνται σε αποστολές διάσωσης. Οι μεγάλες καλλιέργειες μπορούν εύκολα να εποπτεύονται συνεχώς ώστε να αποφασίζεται η περίοδος συγκομιδής. Ολοένα και περισσότεροι άνθρωποι ασχολούνται, ερασιτεχνικά ή επαγγελματικά, με την αεροφωτογραφία, με τα ντοκυμανταίρ και τη δημοσιογραφία και τα drone παίζουν εξέχοντα ρόλο στην εκπλήρωση της αποστολής τους.

Η αποστολή του βίντεο κρυπτογραφημένου χρησιμοποιείται όταν πρέπει η εικόνα του να μην είναι προσβάσιμη από τρίτες οντότητες. Αυτό γίνεται κατά κόρον στις παρακολουθήσεις, στις πολεμικές επιχειρήσεις, στην κατασκοπεία και στην ανακριτική διείσδυση[7] (Περόντση, 2023).

Πέρα από την αποστολή κρυπτογραφημένου βίντεο, η χρησιμότατη βιβλιοθήκη Όρασης Υπολογιστών OpenCV επιτρέπει δυνατότητες που ήταν χρονοβόρες, ασύμφορες ή αδύνατες για τον άνθρωπο. Η ανάγνωση σημάτων κυκλοφορίας και σηματοδοτών επιτρέπει τη δημιουργία μη επανδρωμένων οχημάτων μεταφορά επικίνδυνων υλικών. Η αναγνώριση προσώπων μέσα στο πλήθος μπορεί να βοηθήσει διωκτικές αρχές να εντοπίσουν επικίνδυνους εγκληματίες. Στον ιατρικό κλάδο, μια κάμερα μπορεί να αναγνωρίσει όγκους και ασθένειες ενημερώνοντας συνεχώς για την εξέλιξη κάποιας θεραπείας. Η αυτόνομη Ρομποτική που ανθίζει αλματωδώς ιδιαίτερα τα τελευταία έτη βασίζεται σχεδόν αποκλειστικά σε Computer Vision, ουσιαστικά στην ερμηνεία των εικόνων ώστε να γίνει οποιαδήποτε αντίδραση του συστήματος.

Συνηθως όμως, η υπερβολική εισβολή της τεχνολογίας και μάλιστα παρακολούθησης στην κοινωνία φέρνει αντιδράσεις είτε δεισιδαιμονικές είτε όντως δίκαιες.

7 Η ανακριτική διείσδυση είναι μια εκούσια καταπάτηση κάποιων θεμελιωδών ατομικών δικαιωμάτων. Συνήθως γίνεται με τη σύμφωνη γνώμη των δικαστικών αρχών για την καταπολέμηση γνωστών εγκλημάτων, που δε μπορεί να επιτευχθεί με κανέναν άλλο τρόπο.

ΞΕΝΗ ΒΙΒΛΙΟΓΡΑΦΙΑ

- Adams, J., *"Introducing Raspberry Pi HATs"* (2014 Ιούλιος 31), Ανακτήθηκε από: https://www.raspberrypi.com/news/introducing-raspberry-pi-hats/

- Bate, A., *"Thermal testing Raspberry Pi 4"* (2019 Νοέμβριος 28), Ανακτήθηκε από: https://www.raspberrypi.com/news/thermal-testing-raspberry-pi-4/

- Chen, C., Novick, G., Shimano, K., *"RISC vs CISC"* (2000), Ανακτήθηκε από: https://cs.stanford.edu/people/eroberts/courses/soco/projects/risc/risccisc/

- Comer, D., (2014) "Computer Networks and Internets", Pearson, 6th edition, p. P 346-349.

- Digital Ocean, *"What are the Different Types of Shells in Linux"*, (2022, Αύγουστος 3). Ανακτήθηκε από: https://www.digitalocean.com/community/tutorials/different-types-of-shells-in-linux

- Doole, G. *"Raspberry Pi 4 Pinout, Features and Peripherals"* (2020 Ιούνιος 30), Ανακτήθηκε από: https://microcontrollerslab.com/raspberry-pi-4-pinout-description-features-peripherals-applications/

- Eddison, L., "Coding: Raspberry Pi & Python: A Guide For Beginners", (2018 Ιούνιος 17), CreateSpace Independent Publishing Platform, 2nd edition.

- Intel, *"What is OpenCV"* (2017 Ιούνιος 7), Ανακτήθηκε από: https://www.intel.com/content/www/us/en/developer/articles/technical/what-is-opencv.html

- Khan, A., *"What are Raspberry Pi HATs"* (2022 Φεβρουάριος), Ανακτήθηκε από: https://linuxhint.com/raspberry-pi-hats/

- Lutz, M., (2006) "Programming Python", O'Reilly, 3rd edition, p. P 1032-1035.

- Molloy, D., Exploring Raspberry Pi: Interfacing to the Real World with Embedded Linux, (2016, Ιούνιος 3), Wiley, 1st edition.

- Open Source, *"What is a Raspberry Pi?"* (2022 Νοέμβριος 9), Ανακτήθηκε από: https://opensource.com/resources/raspberry-pi

- Pacheco, P., (2011) "An Introduction to Parallel Programming", Morgan Kaufmann; 1st edition, p. 165

- Pretzel, S., *"Linus Torvalds and Linux"*, Time Toast. Ανακτήθηκε από: https://www.timetoast.com/timelines/linus-torvalds-and-linux-b1880e25-740d-4371-9a09-4d2fd114126d

- Sidhpurwala, H., *"A Brief History of Cryptography"* (2013 Αύγουστος 14), Ανακτήθηκε από: https://www.redhat.com/en/blog/brief-history-cryptography

- Singh, S., The Code Book: The Science of Secrecy from Ancient Egypt to Quantum Cryptography, (2000 Αύγουστος 29), Anchor, 1st edition.

- Software GNU Project, *"About GNU - The first software-sharing community"* (2023, Νοέμβριος 30). Ανακτήθηκε από: https://www.gnu.org/gnu/thegnuproject.html

● Sweidart, A., An Introduction to Building and Breaking Ciphers, (2018 Ιανουάριος), No Starch Press, 1st edition, p.p 312-314.

● Wang, S., "The difference in five modes in the AES encryption algorithm" (2019, Αύγουστος 8). Ανακτήθηκε από: https://www.highgo.ca/2019/08/08/the-difference-in-five-modes-in-the-aes-encryption-algorithm/

● Wikipedia, *"Free Software Foundation"* (2014, Μάιος). Ανακτήθηκε από: https://en.wikipedia.org/wiki/Free_Software_Foundation

● Xingbang, Y., Xuan, P., (2022 n.d.), "Hybrid Technologies for Power Generation", Science Direct, 1st edition, p.p 280-281, 441-442.

ΕΛΛΗΝΙΚΗ ΒΙΒΛΙΟΓΡΑΦΙΑ

- Ελληνική Εταιρεία Ορολογίας (2022 Νοέμβριος), Ανακτήθηκε από: https://eleto.gr/el/orologikoi-poroi/vaseis-oron/anazitisi/

- Κάτος, Β., Στεφανίδης, Γ., "Τεχνικές Κρυπτογραφίας και Κρυπτανάλυσης", 2003 Φεβρουάριος, Εκδόσεις Ζυγός, πρώτη έκδοση, σελ. 11

- Κάτος, Β., Στεφανίδης, Γ., "Τεχνικές Κρυπτογραφίας και Κρυπτανάλυσης", 2003 Φεβρουάριος, Εκδόσεις Ζυγός, πρώτη έκδοση, σελ. 196-200

- Μασχαλίδης, Γ., *"Οσα πρέπει να γνωρίζουν οι ερασιτέχνες χειριστές drone για τη νομοθεσία"* (2017 n.d), Ανακτήθηκε από: https://www.photocontest.gr/articles/osa-prepei-na-gnorizoyn-oi-erasitexnes-xeiristes-drone-gia-ti-nomothesia

- Περόντση Σ., 2023, *"Η ανακριτική διείσδυση"*, Μεταπτυχιακή διατριβή, Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης. Ανακτήθηκε από: https://shorturl.at/ivwAM

- Σαραντάκος Νίκος, *"Drone, δηλαδή δρόνος"* (2017, Ιούνιος 2). Ανακτήθηκε από: https://sarantakos.wordpress.com/2017/06/02/drone/

ΝΟΜΟΘΕΣΙΑ

- Άρθρο 370[Α] παρ. 2 του Ποινικού Κώδικα (Ν.4616/2019), με ενσωματωμένες τις μεταγενέστερες τροποποιήσεις με τις διατάξεις του Ν.4947/2022 - ΦΕΚ 124/Α/23-6-2022.
- Απόφαση Δ/ΥΠΑ/21860/1422/2016 (ΦΕΚ 3152/Β/30-9-2016), "Κανονισμός - γενικό πλαίσιο πτήσεων Συστημάτων μη Επανδρωμένων Αεροσκαφών- ΣμηΕΑ (Unmanned Aircraft Systems – UAS)".
- Κανονισμός 2016/679 του Ευρωπαϊκού Κοινοβουλίου και του Συμβουλίου της 27ης Απριλίου 2016 "για την προστασία των φυσικών προσώπων έναντι της επεξεργασίας των δεδομένων προσωπικού χαρακτήρα και για την ελεύθερη κυκλοφορία των δεδομένων αυτών και την κατάργηση της οδηγίας 95/46/ΕΚ (Γενικός Κανονισμός για την Προστασία Δεδομένων)".
- Ν. 4624/2019 "Αρχή Προστασίας Δεδομένων Προσωπικού Χαρακτήρα", που αντικατέστησε Ν. 2472/1997 "Προστασία του ατόμου από την επεξεργασία δεδομένων προσωπικού χαρακτήρα".
- Ν. 3917/2011 "Διατήρηση δεδομένων που παράγονται ή υποβάλλονται σε επεξεργασία σε συνάρτηση με την παροχή διαθέσιμων στο κοινό υπηρεσιών ηλεκτρονικών επικοινωνιών ή δημόσιων δικτύων επικοινωνιών, χρήση συστημάτων επιτήρησης μετη λήψη ή καταγραφή ήχου ή εικόνας σε δημόσιους χώρους και συναφείς διατάξεις".