ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ
UNIVERSITY OF PATRAS

Πολυτεχνική Σχολή
Τμήμα Μηχανικών Η/Υ & Πληροφορικής

Διπλωματική Εργασία

Ενσωμάτωση Μηχανισμού Κρυπτογράφησης στο Πρωτόκολλο Επικοινωνίας SeedLink

---

Στεφανίδης Μάριος
Α.Μ. 1067458

Επιβλέπων
Σταματίου Ιωάννης
Καθηγητής

Συνεπιβλέπων
Ηλίας Αριστείδης
Ε.ΔΙ.Π.

Πάτρα, 2024

Στους γονείς μου, που με ενέπνευσαν με την υποστήριξη και την τυφλή τους εμπιστοσύνη.

Στον επιβλέποντα καθηγητή μου, κ. Αριστείδη Ηλία., που με καθοδήγησε με την εμπειρία του.

Στους φίλους μου, που ήταν πάντα δίπλα μου σε αυτό το απαιτητικό ταξίδι.

Σε κάθε άνθρωπο που συνέβαλε στην πορεία αυτής της εργασίας.

Περίληψη

Το πρωτόκολλο SeedLink αποτελεί ένα καθολικό πρωτόκολλο επικοινωνίας μεταξύ σεισμολογικών σταθμών και συστημάτων αποθήκευσης/επεξεργασίας σεισμικών δεδομένων σε πραγματικό χρόνο. Χρησιμοποιείται από όλα τα σεισμολογικά ινστιτούτα/εργαστήρια παγκοσμίως για τη συλλογή σεισμολογικών δεδομένων από τους διάφορους σεισμολογικούς σταθμούς και δίκτυα. Ωστόσο, η μεταφορά αυτών των δεδομένων πραγματοποιείται χωρίς κρυπτογράφηση, εκθέτοντας το σύστημα σε πιθανούς κινδύνους όπως κυβερνοεπιθέσεις τύπου "man-in-the-middle" [22][23] και ακολούθως προκαλώντας παραπληροφόρηση από τις αρμόδιες αρχές, όπως είναι η Γενική Γραμματεία Πολιτικής Προστασίας του εκάστοτε κράτους. Στόχος επομένως της παρούσας διπλωματικής εργασίας είναι η υλοποίηση κατάλληλου κρυπτογραφικού μηχανισμού στο πρωτόκολλο επικοινωνίας SeedLink, προκειμένου να διασφαλίζεται η ασφάλεια της μετάδοσης των σεισμολογικών δεδομένων. Αυτό θα αποτρέπει αποτελεσματικά τη δυνατότητα παρακολούθησης, παρεμβολής ή παραποίησης των δεδομένων, προστατεύοντας την ακεραιότητα και την εμπιστευτικότητα των πληροφοριών.

Abstract

The SeedLink protocol constitutes a universal communication protocol between seismological stations and real-time seismic data storage/processing systems. It is utilized by all seismological institutes/laboratories worldwide for collecting seismic data from various seismological stations and networks. However, the transfer of these data occurs without encryption, exposing the system to potential risks such as "man-in-the-middle" cyberattacks [22][23], subsequently leading to misinformation by the relevant authorities, such as the General Secretariat for Civil Protection of the respective state. Therefore, the objective of this thesis is the implementation of an appropriate cryptographic mechanism in the SeedLink communication protocol to ensure the security of seismic data transmission. This will effectively prevent the possibility of monitoring, interfering, or tampering with the data, safeguarding their integrity and confidentiality.

Περιεχόμενα

Λίστα Εικόνων

Λίστα Πινάκων

1

Εισαγωγή

1.1 Σημασία του Προβλήματος

Η κρυπτογραφία των δεδομένων αναδεικνύεται ως θεμέλιο κομμάτι της σύγχρονης ασφάλειας στον ψηφιακό κόσμο. Σε μια εποχή όπου η ποσότητα και η ευαισθησία των πληροφοριών αυξάνεται συνεχώς, η ανάγκη για προστασία των δεδομένων έχει καταστεί κρίσιμη. Η κρυπτογραφία προσφέρει ένα απαραίτητο στρώμα ασφάλειας, προστατεύοντας τις επικοινωνίες, τις συναλλαγές και τις προσωπικές πληροφορίες από ανεπιθύμητη πρόσβαση. Από την προστασία των τραπεζικών συναλλαγών μέχρι τη διασφάλιση του απορρήτου στις επικοινωνίες, η κρυπτογραφία ενισχύει την εμπιστοσύνη και την ασφάλεια σε έναν κόσμο όπου η ψηφιακή επικοινωνία και η αποθήκευση δεδομένων καθίστανται καθημερινή πραγματικότητα.

Όπως η κρυπτογραφία αποτελεί καίριο συστατικό της ψηφιακής ασφάλειας στον ευρύτερο ψηφιακό χώρο, έτσι και στον τομέα της σεισμολογίας η κρυπτογράφηση δεδομένων αναδεικνύεται κρίσιμη, ιδίως όταν χρησιμοποιείται το πρωτόκολλο SeedLink. Η μεταφορά σεισμικών δεδομένων μέσω του SeedLink απαιτεί ένα εξαιρετικό επίπεδο ασφαλείας, καθώς αυτά τα δεδομένα αποτελούν σημαντικό μέρος της γεωεπιστημονικής έρευνας και της πρόβλεψης των σεισμών. Η κρυπτογράφηση εξασφαλίζει την εμπιστευτικότητα και την ακεραιότητα των σεισμικών δεδομένων κατά τη μεταφορά τους, προστατεύοντας τα από πιθανές απειλές ασφάλειας και εξασφαλίζοντας παράλληλα τη σταθερότητα των επιστημονικών αποτελεσμάτων που προκύπτουν από αυτά.

1.2 Στόχος της Διπλωματικής Εργασίας

Η παρούσα διπλωματική εργασία επικεντρώνεται στην ανάπτυξη και υλοποίηση ενός ασφαλούς μηχανισμού κρυπτογράφησης για τα σεισμικά δεδομένα που μεταφέρονται μέσω του πρωτοκόλλου SeedLink. Ο βασικός στόχος είναι η ενίσχυση της ασφάλειας των σεισμογραφικών πληροφοριών κατά τη μετάδοση, προστατεύοντάς τις τελευταίες από πιθανές απειλές ασφαλείας. Ένας από τους κυρίαρχους κινδύνους που προκύπτει από την αλλοίωση αυτών των δεδομένων είναι η πιθανότητα λανθασμένης πληροφόρησης από αρμόδιες αρχές, όπως η Γενική Γραμματεία Πολιτικής Προστασίας.

Πιο συγκεκριμένα, η πιθανότητα παραπλανητικών ή αλλοιωμένων ειδοποιήσεων προς το κοινό από τις πολιτικές αρχές αναδεικνύει την ανάγκη για ένα εξειδικευμένο σύστημα προστασίας. Η κρυπτογράφηση των σεισμικών δεδομένων εγγυάται την εμπιστευτικότητα και την ακεραιότητα των πληροφοριών, μειώνοντας τον κίνδυνο δυνητικών επιθέσεων ή παρεμβάσεων. Επιπλέον, η υιοθέτηση της κρυπτογράφησης δίνει τη δυνατότητα για την επαλήθευση της αυθεντικότητας των ειδοποιήσεων, εμποδίζοντας τη διακίνηση ψευδών πληροφοριών προς το ευρύ κοινό και διασφαλίζοντας την ακρίβεια των ενημερώσεων που λαμβάνουν οι πολίτες. Με αυτόν τον τρόπο, η συγκεκριμένη έρευνα επιδιώκει να συνδυάσει την προηγμένη τεχνολογία κρυπτογράφησης με την ανάγκη για αξιόπιστη, ασφαλή, και ακριβή ενημέρωση στον τομέα της σεισμολογίας.

1.3 Διάρθρωση της Διπλωματικής Εργασίας

Στο κεφάλαιο 2 παρουσιάζεται η ιστορία της κρυπτογραφίας κατά περιόδους. Στη συνέχεια, αναλύονται εκτενώς οι βασικοί ορισμοί και λειτουργίες της, καθώς και οι διάφορες κατηγορίες αλγορίθμων κρυπτογραφίας που υφίστανται. Ειδικότερα, εξετάζεται ο ασύμμετρος κρυπτογραφικός αλγόριθμος RSA, παρέχοντας αναλυτική αναφορά στον τρόπο λειτουργίας και τη σημασία του. Με αυτόν τον τρόπο, προσδίδεται ολοκληρωμένη κατανόηση της εξέλιξης της κρυπτογραφίας και των κρυπτογραφικών μεθόδων.

Στο κεφάλαιο 3 πραγματοποιείται μια εισαγωγή στο μαθηματικό υπόβαθρο που απαιτείται για την κατανόηση του αλγορίθμου AES. Ειδικότερα, αναλύονται λεπτομερώς το πεδίο Galois και ο τρόπος εκτέλεσης των πράξεων σε αυτό, προσφέροντας μια σαφή κατανόηση της μαθηματικής βάσης που υποστηρίζει τον εν λόγω αλγόριθμο.

Στο κεφάλαιο 4, παρουσιάζεται εκτενώς ο τρόπος λειτουργίας του αλγορίθμου AES. Επισημαίνονται λεπτομερώς τα διάφορα βήματα που ακολουθεί κατά την διαδικασία κρυπτογράφησης των δεδομένων. Αυτή η λεπτομερής ανάλυση παρέχει σαφήνεια σχετικά με τη διαδικασία κρυπτογράφησης και ενισχύει την κατανόηση των βημάτων που εμπλέκονται στην προστασία των δεδομένων.

Στο κεφάλαιο 5 πραγματοποιείται λεπτομερής ανάλυση της λειτουργίας του αλγορίθμου GCM. Ειδικότερα, γίνεται αναφορά στα βασικά στοιχεία του αλγορίθμου, ενώ παρέχεται εκτενής ανάλυση του μαθηματικού υποβάθρου που απαιτείται για την κατανόησή του. Εξηγούνται λεπτομερώς οι απαιτήσεις που καθορίζουν τη μοναδικότητα του κλειδιού και του αρχικού διανύσματος (IV), και, σε επέκταση, η σημασία τους για την επίτευξη της ασφάλειας στα δεδομένα.

Στο κεφάλαιο 6 παρουσιάζονται οι βασικές τεχνολογίες που χρησιμοποιήθηκαν όσον αφορά στο υλοποιητικό κομμάτι της παρούσας διπλωματικής εργασίας. Ειδικότερα, παραθέτονται λεπτομερείς πληροφορίες σχετικά με τις τεχνολογίες που επιλέχθηκαν για την υλοποίηση των επιμέρους στοιχείων που αποτελούν τον πυρήνα της εργασίας.

Στο κεφάλαιο 7 παρουσιάζεται λεπτομερώς η υλοποίηση που έχει πραγματοποιηθεί. Ειδικότερα, γίνεται αναφορά στον τυπικό τρόπο με τον οποίο αποκτώνται και μεταφέρονται τα σεισμικά δεδομένα μέσω του πρωτοκόλλου SeedLink και πως αυτή η αρχιτεκτονική τροποποιήθηκε με την ενσωμάτωση της κρυπτογραφικής διαδικασίας. Τέλος, παρουσιάζονται και περιγράφονται λεπτομερώς τα διάφορα επιμέρους στοιχεία που χρησιμοποιήθηκαν στην υλοποίηση, προσφέροντας μία πλήρη επισκόπηση των τεχνολογιών και των διαδικασιών που εφαρμόστηκαν στο πλαίσιο της έρευνας.

Στο κεφάλαιο 8, παρουσιάζεται μία σύνοψη της διπλωματικής εργασίας, επισημαίνοντας τα συμπεράσματα που προέκυψαν μετά από τη διεξοδική έρευνα και την υλοποίηση. Ειδικότερα, αναφέρονται τα κυριότερα ευρήματα που εντοπίστηκαν κατά τη διάρκεια της εξέτασης του θέματος. Επιπλέον, γίνεται αναφορά σε πιθανές αλλαγές που μπορούν να εφαρμοστούν προκειμένου το προτεινόμενο σύστημα να επιτύχει τη βέλτιστη απόδοση.

2

Κρυπτογραφία

Η κρυπτογραφία αποτελεί έναν επιστημονικό κλάδο που ασχολείται κυρίως με την έρευνα, την ανάπτυξη και τη χρήση διαφόρων τεχνικών- εργαλείων κρυπτογράφησης και αποκρυπτογράφησης, με στόχο την απόκρυψη του περιεχομένου των αποσταλέντων μηνυμάτων. Αυτό άλλωστε υποδηλώνουν και τα συνθετικά από τα οποία προέρχεται που είναι “κρυπτός” + “γράφω”.

Πιο συγκεκριμένα, η κρυπτογραφία αναφέρεται στην επιστήμη και στην τέχνη της προστασίας των πληροφοριών από ανεπιθύμητη ή κακόβουλη πρόσβαση ή ακόμα και αλλοίωση μέσω της μετατροπής τους σε μη κατανοητή μορφή, εκτός βέβαια αν κάποιος διαθέτει το κατάλληλο κλειδί αποκρυπτογράφησης. Σκοπός της κρυπτογραφίας αποτελεί η εμπιστευτικότητα, η ακεραιότητα και η διαθεσιμότητα των πληροφοριών. Χρησιμοποιείται ευρέως σε διάφορους τομείς, όπως είναι οι τραπεζικές συναλλαγές, οι επικοινωνίες στο διαδίκτυο, η ασφάλεια των δεδομένων αλλά και σε πολλούς άλλους κλάδους, όπου η προστασία των πληροφοριών είναι ζωτικής σημασίας. Η κρυπτογραφία επιτρέπει την ασφαλή ανταλλαγή πληροφοριών ανάμεσα σε δύο ή περισσότερα μέρη, δημιουργώντας ένα ανθεκτικό προστατευτικό φράγμα έναντι δυνητικών επιθέσεων και απειλών προς την ασφάλεια των πληροφοριών.

Στο παρελθόν, η κρυπτογραφία χρησιμοποιήθηκε για την κρυπτογράφηση μηνυμάτων, δηλαδή τη μετατροπή της πληροφορίας από μία κατανοητή μορφή σε έναν γρίφο, ο οποίος χωρίς τη γνώση του μετασχηματισμού που αξιοποιήθηκε, θα παρέμενε ακατανόητος. Οι παλαιότερες μορφές κρυπτογράφησης βασίζονταν στην επεξεργασία της γλωσσικής δομής. Ωστόσο, στις σύγχρονες μορφές της έμφαση έχει δοθεί ιδιαίτερα σε διάφορα πεδία των μαθηματικών. όπως είναι τα διακριτά μαθηματικά, η θεωρία αριθμών, η θεωρία πληροφορίας, η υπολογιστική πολυπλοκότητα, η στατιστική και η συνδυαστική ανάλυση.

2.1 Η Κρυπτογραφία μέσα στην Ιστορία

2.1.1 Πρώτη Περίοδος Κρυπτογραφίας (1900 π.Χ.- 1900 μ.Χ)

Κατά τη διάρκεια αυτής της χρονικής περιόδου, αναπτύχθηκε ένα εκτενές φάσμα μεθόδων και αλγορίθμων κρυπτογράφησης, τα οποία βασίζονται κατά κύριο λόγο σε απλές αντικαταστάσεις γραμμάτων. Όλες αυτές οι προσεγγίσεις δεν απαιτούσαν εξειδικευμένες γνώσεις ή πολύπλοκες συσκευές, ωστόσο είναι σημαντικό να επισημανθεί η ευφυΐα και ευρηματικότητα των δημιουργών τους. Σε όλα αυτά βέβαια τα συστήματα έχουν γίνει πλέον πολλές αναλύσεις, στις οποίες έχει αποδειχθεί ότι, εάν είναι διαθέσιμο ένα μεγάλο τμήμα του κρυπτογραφημένου μηνύματος, τότε το αρχικό κείμενο μπορεί να ανακτηθεί σχετικά εύκολα.

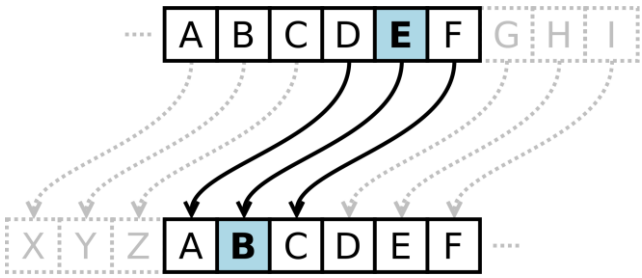
Σύμφωνα με μια μικρή σφηνοειδή επιγραφή που ανακαλύφθηκε στις όχθες του ποταμού Τίγρη, φαίνεται ότι οι πολιτισμοί της Μεσοποταμίας είχαν ήδη ασχοληθεί με την κρυπτογραφία από το 1500 π.Χ. Η συγκεκριμένη επιγραφή περιγράφει μια μέθοδο κατασκευής σμάλτων για αγγειοπλαστική και θεωρείται ως το παλαιότερο κρυπτογραφημένο κείμενο, σύμφωνα με τον Kahn. Επιπλέον, στα Σούσα της Περσίας, ένας άλλος πολιτισμός φαίνεται να δημιούργησε τον αρχαιότερο γνωστό κρυπτοκώδικα στον κόσμο. Σε μια σφηνοειδή επιγραφή, περιέχονται οι αριθμοί 1 έως 8 και από το 32 έως το 35, τοποθετημένοι κατακόρυφα, ενώ απέναντι τους εμφανίζονται τα αντίστοιχα σφηνοειδή σύμβολα.

Οι Σπαρτιάτες, γύρω στον 5ο αιώνα π.Χ., είναι οι πρώτοι που εκμεταλλεύτηκαν την κρυπτογραφία για στρατιωτικούς λόγους. Δημιούργησαν την “σκυτάλη”, η οποία θεωρείται η πρώτη γνωστή κρυπτογραφική συσκευή, χρησιμοποιώντας τη μέθοδο της μετάθεσης για την κρυπτογράφηση. Σύμφωνα με τον Πλούταρχο, η “Σπαρτιατική Σκυτάλη” (βλ. Εικόνα 1) είχε τη μορφή μιας ξύλινης ράβδου, ορισμένης διαμέτρου, γύρω από την οποία τυλιγόταν ελικοειδώς μια λωρίδα περγαμηνής. Το κείμενο ήταν οργανωμένο σε στήλες, με ένα γράμμα ανά έλικα. Όταν ξετυλίγονταν η λωρίδα, το κείμενο γινόταν ακατάληπτο λόγω της αναδιάταξης των γραμμάτων. Το “κλειδί” της κρυπτογραφίας ήταν η διάμετρος της σκυτάλης.

Εικόνα 1: *Scytale*

Στην αρχαιότητα, οι καταγραφές και τα συστήματα επικοινωνίας βασίζονταν κυρίως στη στεγανογραφία παρά στην κρυπτογραφία. Αν και οι αρχαίοι Έλληνες συγγραφείς δεν καταγράφουν εκτενώς τη χρήση συστημάτων γραπτής αντικατάστασης γραμμάτων, αυτά εμφανίζονται στους Ρωμαίους, ιδιαίτερα κατά την περίοδο του Ιουλίου Καίσαρα. Ο Καίσαρας χρησιμοποίησε το σύστημα κρυπτογράφησης που στηρίζεται στην αντικατάσταση των γραμμάτων του αλφαβήτου με άλλα που βρίσκονται σε συγκεκριμένες θέσεις πριν ή μετά, γνωστό ως "κρυπτοσύστημα του Καίσαρα". Ο Καίσαρας εφάρμοσε επίσης πιο πολύπλοκα συστήματα κρυπτογράφησης. για τα οποία γράφτηκε βιβλίο από τον Valerius Probus. Δυστυχώς, το συγκεκριμένο βιβλίο δεν διασώθηκε αλλά θεωρείται το πρώτο βιβλίο κρυπτολογίας παρά την απώλειά του. Το σύστημα αντικατάστασης του Καίσαρα χρησιμοποιήθηκε εκτενώς και στους επόμενους αιώνες, ενισχύοντας τη σημασία του ως ένα από τα πρώτα γνωστά κρυπτογραφικά συστήματα.

Στην Εικόνα 2 παρουσιάζεται η διαδικασία κρυπτογράφησης που πρότεινε ο Ιούλιος Καίσαρας. Πιο συγκεκριμένα, χρησιμοποιείται μια αριστερή μετατόπιση κατά τρεις θέσεις, έτσι ώστε για παράδειγμα κάθε εμφάνιση του γράμματος Ε στο απλό κείμενο (plaintext) να γίνεται Β στο κρυπτογραφημένο (cipher text).



Εικόνα 2: *Caesar Cipher*

Κατά τη διάρκεια του Μεσαίωνα, η κρυπτολογία θεωρούνταν απαγορευμένη και συνδεόταν με τη μαύρη μαγεία, καθυστερώντας την ανάπτυξή της. Η εξέλιξη της κρυπτολογίας και των μαθηματικών συνεχίστηκε στον Αραβικό κόσμο, όπως αποτυπώνεται στις "Χίλιες και μία νύχτες" με λέξεις-αινίγματα. Οι Άραβες εισήγαγαν κρυπταλφάβητα, όπως το "Dawoudi", ενώ ήταν οι πρώτοι που έκαναν χρήση μεθόδων κρυπτανάλυσης. Η αξιοποίηση των συχνοτήτων των γραμμάτων σε συνδυασμό με τις γλωσσικές συχνότητες εμφάνισης εφευρέθηκε γύρων στον 14ο αιώνα. Η κρυπτογραφία εξελίχθηκε στους επόμενους αιώνες με πρωτοπόρους όπως ο Giovanni Batista Porta, που δημοσίευσε το "De furtivis literarum notis" το 1563, παρουσιάζοντας πολυαλφαβητικά συστήματα και διγραφικά κρυπτογραφήματα. Ο Γάλλος Vigenere είναι ένας σημαντικός εκπρόσωπος της εποχής, με τον πίνακα πολυαλφαβητικής αντικατάστασης που χρησιμοποιείται ακόμη και σήμερα.

Ο C. Wheatstone, γνωστός για τις μελέτες του στον ηλεκτρισμό, παρουσίασε την πρώτη μηχανική κρυπτοσυσκευή, η οποία υπηρέτησε ως βάση για την εξέλιξη των κρυπτομηχανών στη δεύτερη ιστορική περίοδο της κρυπτογραφίας. Η αποκρυπτογράφηση των αιγυπτιακών ιερογλυφικών, που παρέμεναν μυστήρια για αιώνες, επιτεύχθηκε μέσω κρυπταναλυτικής εργασίας. Τα ιερογλυφικά, που χρονολογούνται από το 3000 π.Χ., ήταν υπερβολικά πολύπλοκα, με αποτέλεσμα παράλληλα με αυτά να αναπτυχθεί και η ιερατική γραφή με σκοπό την καθημερινή χρήση. Στον 17ο αιώνα, το ενδιαφέρον για την αποκρυπτογράφηση αυξήθηκε, με τον Κίρχερ να εκδίδει το ‘Oedipus Aegyptiacus” το 1652. Παρά την αποτυχημένη προσπάθειά του, η ανακάλυψη της “Στήλης της Ροζέτας” άνοιξε το δρόμο για τη σωστή ερμηνεία των ιερογλυφικών, με τους Γιανγκ και Σαμπολιόν να καταφέρνουν την αποκρυπτογράφηση τους. Οι προϊστορικοί πληθυσμοί χρησιμοποίησαν τρεις γραφές πριν την επινόηση του αλφαβήτου, περίπου το 850 π.Χ.

Χρονολογικά, οι γραφές αυτές κατατάσσονται ως εξής:

- 3000- 1600 π.Χ. : Εικονογραφική (Ιερογλυφική) γραφή
- 1850- 1450 π.Χ.: Γραμμική γραφή Α
- 1450- 1200 π.Χ.: Γραμμική Γραφή Β

Η κρητική εικονογραφική γραφή, γνωστή και ως ιερογλυφική, παραμένει κρυπτογραφημένη, αφού δεν έχει αποκαλυφθεί ο κώδικάς της. Παρόλο που γνωρίζουμε ότι δεν χρησιμοποιεί εικόνες ως σημεία, αλλά αντιπροσωπεύει φωνήεντα, ανασκαφές στο ανάκτορο των Μαλίων στην Κρήτη αποκαλύπτουν περίπου 200 σφραγιδόλιθους που φέρουν αυτή τη γραφή, συνυπάρχοντας με τη γραμμική γραφή Α. Στον δίσκο της Φαιστού (βλ. **Εικόνα 3**), που χρονολογείται στο 1700 π.Χ. εμφανίζονται δύο σπείρες συμβόλων που χαράχτηκαν με σφραγίδες, καθιστώντας τον Δίσκο το αρχαιότερο δείγμα στοιχειοθεσίας. Αν και παραμένει κρυπτογραφημένος, η Κρητική εικονογραφική γραφή παραμένει η πιο μυστηριώδης αρχαία ευρωπαϊκή γραφή.

Εικόνα 3: *Phaistos Disk*

Ο Σερ Άρθουρ Έβανς, μεγάλος Άγγλος αρχαιολόγος, ανακάλυψε τις πρώτες επιγραφές με Γραμμική γραφή κατά την ανασκαφή της Κνωσού το 1900. Την ονόμασε "γραμμική" εξαιτίας του γεγονότος ότι τα γράμματα της αποτελούν γραμμές, σε αντίθεση με τη σφηνοειδή γραφή ή τις εικόνες της αιγυπτιακής ιερατικής. Πιθανότατα, η γραμμική γραφή Α ανήκει στους Μινωίτες, κατοίκους της αρχαίας Κρήτης, και μπορεί να αποτελεί τον πρόδρομο του σύγχρονου ελληνικού αλφαβήτου. Τα γράμματα χαράζονταν σε πήλινες πλάκες με χρήση κάποιου αιχμηρού αντικειμένου και στη συνέχεια ξεραίνονταν σε φούρνους. Οι περισσότερες επιγραφές (περίπου 1500) είναι λογιστικές, περιλαμβάνοντας εικόνες ή συντομογραφίες προϊόντων και αριθμούς για την υπόδειξη ποσότητας ή οφειλής.

Ο Έβανς κατέγραψε 135 σύμβολα της Γραμμικής Α, που χρησιμοποιήθηκε στην Κρήτη. Πρόσφατα ευρήματα υποδεικνύουν χρήση και αλλού, όπως σε Κνωσό, Φαιστό, Μήλο, και Θήρα ενώ πλάκες με επιγραφές σε γραμμική Α εκτίθενται στο Μουσείο Ηρακλείου. Παρά την τεράστια πρόοδο, ακόμη και σήμερα η αποκρυπτογράφηση της Γραμμικής Α παραμένει ανεπίλυτη. Ο Evans ονόμασε τη Γραμμική Γραφή Β, αναγνωρίζοντας την ως συγγενική με τη γραμμική Α αν και πιο πρόσφατη, ενώ φαίνεται πως υιοθετήθηκε αποκλειστικά για λογιστικούς σκοπούς. Πινακίδες με τη γραφή αυτή βρέθηκαν σε πολλές τοποθεσίες, συνολικά περίπου στα 10.000 τεμάχια. Τα σχήματα αυτών των πινακίδων ποικίλουν, αλλά επικρατούν κυρίως οι φυλλοειδείς και «σελιδόσχημες». Κατασκευάζονταν από πηλό, πιέζονταν μέχρι να γίνουν επίπεδες, επιμήκης και συμπαγής πινακίδες. Πολλές φορές σχημάτιζαν "ομάδες" ή "πολύπτυχα" πινακίδων για μακροσκελή κείμενα, φυλασσόμενα σε αρχειοφυλάκια. Ο Βέντρις, ενδιαφερόμενος αρχαιολόγος, πρώτος αναγνώρισε τη γραφή ως ελληνική, αν και αρχικά η άποψή του δεν έγινε αποδεκτή. Η κρυπτανάλυση της Γραμμικής Β το 1953, με τη συνδρομή του Τσάντγουικ, απέδειξε ότι επρόκειτο για ελληνική γλώσσα καθώς και τη δυνατότητα των Μινωιτών να μιλούν ελληνικά.

2.1.2 Δεύτερη Περίοδος Κρυπτογραφίας (1900- 1950)

Η δεύτερη περίοδος της κρυπτογραφίας, που εκτείνεται από τις αρχές του 20ού αιώνα έως το 1950, επηρεάζεται κυρίως από τους παγκόσμιους πολέμους. Κατά τη διάρκεια αυτής της περιόδου, η ανάγκη για ασφάλεια στη μετάδοση ζωτικών πληροφοριών οδηγεί στην ανάπτυξη πολύπλοκων κρυπτοσυστημάτων, συμπεριλαμβανομένων των κρυπτομηχανών. Η ανάλυση αυτών των συστημάτων απαιτεί μεγάλο προσωπικό και υπολογιστική ισχύ. Παρά την πολυπλοκότητά τους, η κρυπτανάλυση συνήθως επιτυγχάνεται, όπως φαίνεται στη χρήση του γερμανικού συστήματος Enigma κατά τη διάρκεια του Β' Παγκοσμίου Πολέμου.

Εικόνα 4: *Enigma*

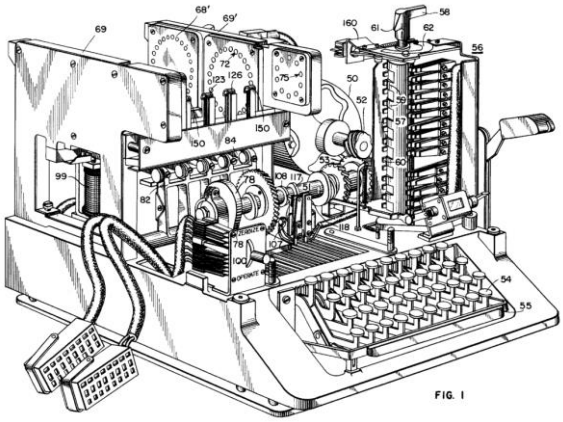
Ο Μαριάν Ρέγεφσκι στην Πολωνία παραβίασε την πρώτη έκδοση του γερμανικού κρυπτογραφικού συστήματος Enigma με θεωρητικές μαθηματικές μεθόδους το 1932. Αυτή η επιτυχία ήταν καθοριστική όσον αφορά στην κρυπτολογική ανάλυση της εποχής. Οι Πολωνοί συνέχισαν να αποκρυπτογραφούν μηνύματα με Enigma μέχρι το 1939, όταν οι Γερμανοί έκαναν αλλαγές που υπερέβαιναν τις δυνατότητές τους. Μετέφεραν τις γνώσεις τους μαζί με κάποιες μηχανές που είχαν κατασκευάσει στους Βρετανούς και τους Γάλλους. Αυτό είχε ως αποτέλεσμα ο Rejewski και οι μαθηματικοί και κρυπτογράφοι του, όπως ο Biuro Szyfrow να συνεργαστούν με τους Βρετανούς και Γάλλους. Η παραπάνω συνεργασία συνεχίστηκε από τον Alan Turing. τον Γκόρντον Ουέλτσμαν και από πολλούς άλλους. η οποία οδήγησε σε συνεχείς αποκρυπτογραφήσεις με τη χρήση του υπολογιστή Colossus. Οι αμερικανοί κρυπτογράφοι, σε συνεργασία με Βρετανούς και Ολλανδούς, αποκρυπτογράφησαν κρυπτοσυστήματα του Ιαπωνικού ναυτικού, συμβάλλοντας στην αμερικανική επιτυχία στη Ναυμαχία της Μιντγουέι και την εξόντωση του Ιαπωνικού Στόλου.

Το Ιαπωνικό Υπουργείο Εξωτερικών χρησιμοποίησε το κρυπτογραφικό σύστημα "Purple," το οποίο ανέπτυξε τοπικά, καθώς και διάφορες μηχανές για τις συνδέσεις ιαπωνικών πρεσβειών. Μία από αυτές, η "Μηχανή-Μ," και μια άλλη που αναφέρθηκε ως "Red," ήταν γνωστές στις ΗΠΑ. Η ομάδα SIS του αμερικανικού στρατού κατάφερε να αποκρυπτογραφήσει το ιαπωνικό διπλωματικό σύστημα κρυπτογράφησης "Purple," μια ηλεκτρομηχανική συσκευή, πριν από την έναρξη του Β' Παγκοσμίου

Πολέμου. Το αποτέλεσμα αυτής της κρυπτανάλυσης, ειδικότερα της "Purple," αναφερόταν από τους Αμερικανούς ως "Magic" (Μαγεία).

Στον Δεύτερο Παγκόσμιο Πόλεμο, οι συμμαχικές κρυπτομηχανές συμπεριλάμβαναν το βρετανικό TypeX και το αμερικανικό SIGABA. Και τα δύο ήταν ηλεκτρομηχανικά σχέδια παρόμοια με το Enigma, αλλά με σημαντικές βελτιώσεις. Δεν υπήρξε καμία γνωστή περίπτωση παραβίασης τους κατά τη διάρκεια του πολέμου. Στο πεδίο μάχης, χρησιμοποιήθηκαν οι κρυπτομηχανές M-209 και η λιγότερο ασφαλής οικογένεια M-94. Οι Βρετανοί πράκτορες της υπηρεσίας "SOE" αρχικά χρησιμοποίησαν ένα τύπο κρυπτογραφίας βασισμένο σε ποιήματα, όπου τα ποιήματα λειτουργούσαν ως κλειδιά. Παρά τη σύλληψη ενός ποιήματος του Πολ Βερλέν από τους Γερμανούς πριν την Απόβαση της Νορμανδίας, δεν κατάφεραν να το αποκρυπτογραφήσουν και δεν έλαβαν σοβαρά υπόψη τους την προειδοποίηση.

Οι Πολωνοί είχαν προετοιμαστεί για την εμπόλεμη περίοδο κατασκευάζοντας την κρυπτομηχανή LCD Lacida, η οποία κρατήθηκε μυστική ακόμη και από τον Rejewski. Τον Ιούλιο του 1941, όταν ο Rejewski ελέγχθηκε για την ασφάλειά της, κατάφερε να την "σπάσει" μέσα σε λίγες ώρες, αναγκάζοντας τους Πολωνούς να την αλλάξουν άμεσα. Τα μηνύματα που στάλθηκαν με την Lacida δεν ήταν εξίσου πολύπλοκα με αυτά του Enigma, αλλά η παρεμπόδιση της μηχανής θα μπορούσε να σηματοδοτήσει το τέλος της κρίσιμης πολωνικής κρυπταναλυτικής προσπάθειας.



Εικόνα 5: *Sigaba*

2.1.3 Τρίτη Περίοδος Κρυπτογραφίας (1950- σήμερα)

Η περίοδος αυτή χαρακτηρίζεται από την άνθηση της ανάπτυξης στους επιστημονικούς κλάδους των μαθηματικών, της μικροηλεκτρονικής και των υπολογιστικών συστημάτων. Η εποχή της σύγχρονης κρυπτογραφίας ξεκινά ουσιαστικά με τον Claude Shannon, που θεωρείται ο πατέρας των μαθηματικών συστημάτων κρυπτογραφίας. Το 1949, δημοσίευσε το έγγραφο "Θεωρία επικοινωνίας των συστημάτων μυστικότητας" στο τεχνικό περιοδικό Bell System και αργότερα το βιβλίο "Μαθηματική Θεωρία της

Επικοινωνίας" μαζί με τον Warren Weaver. Αυτές οι εργασίες θεμελίωσαν μια στερεά θεωρητική βάση για την κρυπτογραφία και την κρυπτανάλυση. Κατά τη διάρκεια αυτής της εποχής, η κρυπτογραφία εξαφανίζεται και φυλάσσεται από τις μυστικές υπηρεσίες κυβερνητικών επικοινωνιών, όπως η NSA. Ελάχιστες καινοτομίες δημοσιοποιήθηκαν ξανά μέχρι τα μέσα της δεκαετίας του '70, όταν πραγματοποιήθηκαν σημαντικές αλλαγές.

Στα μέσα της δεκαετίας του '70 σημειώθηκαν δύο σημαντικές πρόοδοι στον τομέα της κρυπτογραφίας. Πρώτον, το σχέδιο προτύπου κρυπτογράφησης DES (Data Encryption Standard) δημοσιεύτηκε στον ομοσπονδιακό κατάλογο της Αμερικής στις 17 Μαρτίου 1975. Το DES προτάθηκε από την IBM ως απάντηση της πρόσκλησης του Εθνικού Γραφείου Προτύπων (NIST) με σκοπό την ανάπτυξη ασφαλών ηλεκτρονικών συστημάτων επικοινωνίας για επιχειρήσεις, όπως τράπεζες και άλλες οικονομικές οργανώσεις. Μετά από συμβουλές και τροποποιήσεις από την NSA, το πρότυπο αυτό υιοθετήθηκε και δημοσιεύθηκε ως ένα ομοσπονδιακό πρότυπο επεξεργασίας πληροφοριών το 1977. Ο DES ήταν ο πρώτος δημόσια προσιτός αλγόριθμος κρυπτογράφησης που εγκρίθηκε από μια εθνική αρχή όπως η NSA, προκαλώντας θύελλα ενδιαφέροντος από τη δημόσια και ακαδημαϊκή κοινότητα για τα κρυπτογραφικά συστήματα.

Ο DES (Data Encryption Standard) αντικαταστάθηκε επίσημα από τον AES (Advanced Encryption Standard) το 2001, όταν το Εθνικό Ινστιτούτο Προτύπων και Τεχνολογίας (NIST) ανακοίνωσε το FIPS 197. Μετά από έναν ανοικτό διαγωνισμό, το NIST επέλεξε τον αλγόριθμο Rijndael, υποβληθείς από δύο Φλαμανδούς κρυπτογράφους, για να γίνει ο AES. Πιο συγκεκριμένα, ο DES, μαζί με ασφαλέστερες παραλλαγές του, όπως ο 3DES ή TDES, χρησιμοποιείται ακόμη και σήμερα, ενσωματωμένος σε πολλά εθνικά και οργανωτικά πρότυπα. Παρόλα αυτά, το βασικό μέγεθος των 56 bit έχει αποδειχθεί ανεπαρκές να αντισταθεί σε επιθέσεις ωμής βίας, με τον DES να έχει “εξουδετερωθεί” σε σχετικά σύντομο χρονικό διάστημα. Η χρήση απλής κρυπτογράφησης με τον DES θεωρείται επισφαλής, και όλα τα μηνύματα που έχουν αποσταλεί από το 1976 με χρήση DES ενδέχεται να κινδυνεύουν από αποκρυπτογράφηση. Το γεγονός αυτό οδήγησε στην ανάπτυξη πιο ασφαλών αλγορίθμων κρυπτογράφησης όπως είναι ο AES.

2.2 Βασικές Λειτουργίες Κρυπτογραφίας

Η κρυπτογραφία αποτελεί ένα θεμελιώδες εργαλείο για την ασφάλεια πληροφοριών και των επικοινωνιών στον ψηφιακό κόσμο. Οι βασικές λειτουργίες της κρυπτογραφίας είναι η εμπιστευτικότητα, η ακεραιότητα, η μη απάρνηση και η πιστοποίηση. Πιο συγκεκριμένα:

- **Εμπιστευτικότητα (Confidentiality)**- αναφέρεται στη δυνατότητα προστασίας των πληροφοριών από μη εξουσιοδοτημένη πρόσβαση. Η κρυπτογραφία επιτυγχάνει αυτή τη

λειτουργία με τον τρόπο που μετατρέπει τα δεδομένα σε μια ακατανόητη μορφή για όποιον δεν διαθέτει το κατάλληλο κλειδί αποκρυπτογράφησης. Κατά αυτό τον τρόπο, μόνο οι εξουσιοδοτημένοι χρήστες που διαθέτουν το κατάλληλο κλειδί μπορούν να αποκρυπτογραφήσουν τα δεδομένα και να τα διαβάσουν. Η εμπιστευτικότητα γενικά αναφέρεται στο περιεχόμενο ηλεκτρονικών εγγράφων, αρχείων και μηνυμάτων καθώς και στη φύση τους και στην ταυτότητα εκείνων που εκτελούν ενέργειες και ανταλλάσσουν πληροφορίες. Επιπλέον, αναφέρεται στο χρόνο και στην ποσότητα των μηνυμάτων που ανταλλάσσονται. Η εμπιστευτικότητα, σε ορισμένες περιπτώσεις, ορίζεται και ως "ιδιωτικότητα" ή "μυστικότητα" ή "προστασία απορρήτου".

- **Ακεραιότητα (Integrity)-** επιδιώκει τη διατήρηση της ακριβούς μορφής των δεδομένων χωρίς καμία τροποποίηση από μη εξουσιοδοτημένα μέρη. Η κρυπτογραφία εξασφαλίζει την ακεραιότητα με τη χρήση ψηφιακών υπογραφών και σφραγίδων που επιβεβαιώνουν ότι τα δεδομένα προέρχονται από συγκεκριμένο πηγαίο επίπεδο και δεν έχουν τροποποιηθεί κατά τη μεταφορά τους. Έχουν αναγνωριστεί τρεις καθοριστικές συνιστώσες του όρου ακεραιότητα: οι "εξουσιοδοτημένες ενέργειες", ο "διαχωρισμός και η προστασία αγαθών" και τέλος "η ανίχνευση και διόρθωση σφαλμάτων".
- **Μη Απάρνηση (Non- Repudiation)**- σημαίνει ότι ένας χρήστης δεν μπορεί να αρνηθεί τη συμμετοχή του σε μια επικοινωνία ή τη δημιουργία ενός μηνύματος. Η κρυπτογραφία διασφαλίζει τη μη απάρνηση μέσω της ψηφιακής υπογραφής, η οποία συνδέει μοναδικά ένα μήνυμα με τον αποστολέα του, επιτρέποντας την αναγνώριση της προέλευσής του.
- **Πιστοποίηση (Authentication)**- επιβεβαιώνει την αυθεντικότητα των χρηστών ή των πόρων στο διαδίκτυο. Η κρυπτογραφία χρησιμοποιείται για τη δημιουργία ψηφιακών πιστοποιητικών, τα οποία επιβεβαιώνουν την ταυτότητα και την αξιοπιστία των συμμετεχόντων σε μια επικοινωνία ή σε μια διαδικασία.

2.3 Βασικοί Ορισμοί Κρυπτογραφίας

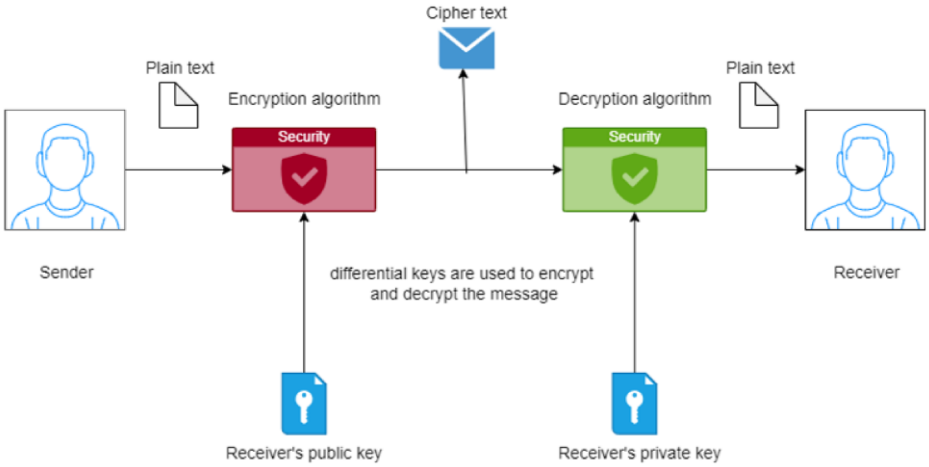
Παρακάτω παρουσιάζονται αναλυτικοί ορισμοί που αφορούν στις βασικές έννοιες του τομέα της κρυπτογραφίας.

1. **Κρυπτογράφηση (Encryption)**- ασχολείται με τη διαδικασία μετατροπής ενός αρχικού κειμένου (plaintext) σε μία ακατανόητη μορφή ή διαφορετικά σε ένα κρυπτογραφημένο κείμενο (ciphertext) μέσω μιας κρυπτογραφικής λειτουργίας, συνήθως με τη χρήση ενος κλειδιού.

2. **Αποκρυπτογράφηση (Decryption)**- είναι η αντίστροφη διαδικασία της κρυπτογράφησης, δηλαδή η μετατροπή του κρυπτογραφημένου κειμένου (ciphertext) πίσω στην αρχική του μορφή (plaintext), χρησιμοποιώντας το κατάλληλο κλειδί.
3. **Αρχικό Κείμενο (Plaintext)**- αναφέρεται στα αρχικά δεδομένα ή το μήνυμα που χρειάζεται να μεταδοθεί. Είναι ουσιαστικά το κείμενο που υπόκειται στη διαδικασία της κρυπτογράφησης.
4. **Κλειδί (Key)**- αποτελεί μια μοναδική ακολουθία ψηφίων (ακολουθία bit) ή ένα σύνολο κανόνων που χρησιμοποιείται από τον κρυπτογραφικό αλγόριθμο για τη μετατροπή του αρχικού κειμένου σε κρυπτογραφημένο κείμενο και αντίστροφα για την αποκρυπτογράφηση.
5. **Κρυπτογραφημένο Κείμενο (Ciphertext)**- ορίζεται ως το αποτέλεσμα της διαδικασίας κρυπτογράφησης. Είναι η μορφή των δεδομένων που προκύπτει μετά την εφαρμογή του κρυπτογραφικού αλγορίθμου στο αρχικό κείμενο σε συνδυασμό με τη χρήση του κλειδιού.
6. **Κρυπτανάλυση (Cryptanalysis)**- είναι μία επιστήμη που ασχολείται με το "σπάσιμο" κάποιας κρυπτογραφικής τεχνικής, ώστε ακόμη και όταν το κλειδί δεν είναι γνωστό, η αποκωδικοποίηση του αρχικού κειμένου καθίσταται δυνατή.
7. **Κρυπτογραφικός Αλγόριθμος (Cipher)**- αναφέρεται ως ένας μαθηματικός προσδιορισμός ή διαδικασία που χρησιμοποιείται για την εκτέλεση της κρυπτογράφησης και της αποκρυπτογράφησης. Αποτελεί ουσιαστικά τη μέθοδο μετασχηματισμού δεδομένων σε μία μορφή που δεν επιτρέπει την αποκάλυψη των περιεχομένων τους από μη εξουσιοδοτημένα μέλη.

Γενικότερα, η διαδικασία της κρυπτογράφησης και αποκρυπτογράφησης ενός μηνύματος βασίζεται σε έναν αλγόριθμο κρυπτογράφησης (cipher) και ένα κλειδί κρυπτογράφησης (key). Συνήθως, ο αλγόριθμος κρυπτογράφησης είναι γνωστός, επομένως η εμπιστευτικότητα του κρυπτογραφημένου μηνύματος εξαρτάται κυρίως από την μυστικότητα του κλειδιού. του οποίου το μέγεθος μετριέται σε αριθμό bits. Γενικά, ισχύει η αρχή ότι όσο μεγαλύτερο είναι το κλειδί, τόσο δυσκολότερο είναι να αποκρυπτογραφηθεί το κρυπτογραφημένο μήνυμα από κακόβουλα τρίτα μέλη. Επίσης, διάφοροι αλγόριθμοι κρυπτογράφησης απαιτούν διαφορετικά μήκη κλειδιών για να επιτύχουν το ίδιο επίπεδο ασφάλειας.

Στην **Εικόνα 6** φαίνεται πως όλες οι παραπάνω έννοιες συνδέονται μεταξύ τους.



Εικόνα 6: *Encryption- Decryption System*

2.4 Σύστημα Κρυπτογράφησης- Αποκρυπτογράφησης

Ο αντικειμενικός σκοπός της κρυπτογραφίας είναι να διευκολύνει την ασφαλή επικοινωνία μεταξύ δύο ατόμων, όπως ο Κώστας και η Βασιλική, μέσω ενός μη ασφαλούς καναλιού. Ο στόχος είναι να εξασφαλίσει ότι ένα τρίτο, μη εξουσιοδοτημένο πρόσωπο, όπως ένα κακόβουλο πρόσωπο, δεν μπορεί να παρεμβληθεί στην επικοινωνία ή να κατανοήσει το περιεχομένου των μηνυμάτων.

Ένα κρυπτοσύστημα (σύνολο διαδικασιών κρυπτογράφησης- αποκρυπτογράφησης) αποτελείται από μία πεντάδα (P, C ,k, E, D):

1. **Πλαίσιο (P- Plaintext)**- αναφέρεται στα δεδομένα που είναι ανοιχτά και προσπελάσιμα χωρίς κρυπτογράφηση. Στη διαδικασία κρυπτογράφησης, αυτά τα δεδομένα μετατρέπονται σε κρυπτοκείμενο.
2. **Κρυπτοκείμενο (C- Ciphertext)**- είναι το αποτέλεσμα της κρυπτογράφησης των δεδομένων πλαισίου (plaintext). Είναι τα δεδομένα που έχουν υποστεί μετατροπή και πλέον είναι δυσανάγνωστα χωρίς το κατάλληλο κλειδί.
3. **Κλειδί (k- Key)**- το κλειδί είναι ένας αλγόριθμος ή μια σειρά από ψηφία που χρησιμοποιούνται για την κρυπτογράφηση και αποκρυπτογράφηση των δεδομένων. Είναι ουσιαστικά η "συνταγή" που εφαρμόζεται στα δεδομένα.
4. **Διαδικασία Κρυπτογράφησης (E- Encryption)**- αναφέρεται στον κρυπτογραφικό μετασχηματισμό ή τη κρυπτογραφική συνάρτηση που εφαρμόζεται για την μετατροπή των δεδομένων σε κρυπτοκείμενο, χρησιμοποιώντας το κλειδί.

5. **Διαδικασία Αποκρυπτογράφησης (D- Decryption)**- είναι η αντίστροφη διαδικασία της κρυπτογράφησης. Χρησιμοποιείται το ίδιο κλειδί για να αποκρυπτογραφήσει το κρυπτοκείμενο, επαναφέροντας τα δεδομένα στην αρχική τους μορφή.

Η συνάρτηση κρυπτογράφησης E δέχεται δύο παραμέτρους από το χώρο P και το χώρο k παράγοντας μία ακολουθία που ανήκει στο χώρο C. Συγκεκριμένα, αυτή η συνάρτηση εφαρμόζει έναν αλγόριθμο κρυπτογράφησης για να μετασχηματίσει τα δεδομένα από το χώρο P σε μια αντιστοιχία στο χώρο C. Αντίστοιχα, η συνάρτηση αποκρυπτογράφησης D δέχεται δύο παραμέτρους, τον χώρο C και το χώρο k, και παράγει μια ακολουθία που ανήκει στο χώρο P. Αυτή η συνάρτηση εφαρμόζει τον αλγόριθμο αποκρυπτογράφησης, για να αντιστρέψει τη διαδικασία και να επαναφέρει τα δεδομένα στον αρχικό χώρο P.

Ένα τέτοιο σύστημα λειτουργεί ως εξής:

1. **Επιλογή Κλειδιού**- ο αποστολέας επιλέγει ένα κλειδί μήκους n από το χώρο κλειδιών με τυχαίο τρόπο. Τα n στοιχεία του κλειδιού K ανήκουν σε ένα πεπερασμένο αλφάβητο.
2. **Ασφαλής Αποστολή του Κλειδιού**- το κλειδί αποστέλλεται στον παραλήπτη μέσω ενός ασφαλούς καναλιού
3. **Δημιουργία Μηνύματος**- ο αποστολέας δημιουργεί ένα μήνυμα από το χώρο των μηνυμάτων
4. **Κρυπτογράφηση Μηνύματος**- η συνάρτηση κρυπτογράφησης λαμβάνει το κλειδί και το μήνυμα ως είσοδο και παράγει μια κρυπτοακολουθία συμβόλων, γνωστή ως γρίφος. Αυτή η κρυπτοακολουθία αποστέλλεται μέσω ενός μη ασφαλούς καναλιού.
5. **Αποστολή του Γρίφου**- η κρυπτοακολουθία, δηλαδή ο γρίφος, αποστέλλεται μέσω του μη ασφαλούς καναλιού προς τον παραλήπτη.
6. **Αποκρυπτογράφηση**- η συνάρτηση αποκρυπτογράφησης λαμβάνει το κλειδί και το γρίφο ως είσοδο και παράγει την ισοδύναμη ακολουθία μηνύματος.

Στην ουσία, το σύστημα εξασφαλίζει ότι μόνο ο παραλήπτης που διαθέτει το κατάλληλο κλειδί, μπορεί να αποκρυπτογραφήσει το μήνυμα και να ανακτήσει την αρχική πληροφορία.

Στην περίπτωση που ο αντίπαλος παρακολουθεί την επικοινωνία αλλά δεν διαθέτει το κλειδί για την αποκρυπτογράφηση, η ασφάλεια του συστήματος βασίζεται στην δυσκολία εξαγωγής του κλειδιού από την κρυπτοακολουθία. Επιλέγοντας να παρακολουθεί όλα τα μηνύματα, ο αντίπαλος εστιάζει στην προσπάθεια ανακάλυψης του κλειδιού για αποκρυπτογράφηση οποιουδήποτε μηνύματος. Αυτό μπορεί να απαιτήσει σημαντικό χρόνο και πόρους, ειδικά εάν το κλειδί είναι αρκετά μεγάλο και πολύπλοκο.

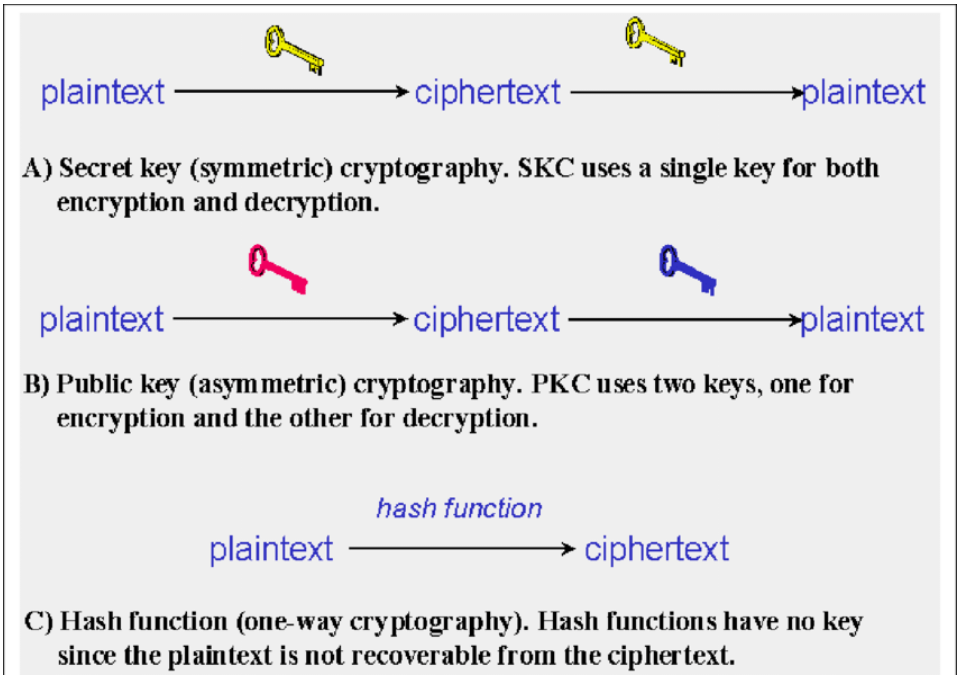
Αν ο αντίπαλος ενδιαφέρεται μόνο για το υπάρχον μήνυμα, θα προσπαθήσει να παράγει μια εκτίμηση για την πληροφορία του μηνύματος χωρίς να αποκρυπτογραφήσει το κείμενο πλήρως. Αυτή η

διαδικασία, γνωστή ως κρυπτανάλυση, μπορεί να παράγει προσεγγίσεις ή εκτιμήσεις για το περιεχόμενο του μηνύματος, αλλά δεν εξασφαλίζει την ακριβή ανακατασκευή του αρχικού μηνύματος χωρίς το κλειδί.

2.5 Τύποι Αλγορίθμων Κρυπτογραφίας

Υπάρχουν ποικίλοι τρόποι ταξινόμησης των κρυπτογραφικών αλγορίθμων. Σε αυτό το κεφάλαιο θα γίνει διακριτή κατηγοριοποίησή τους βάσει του αριθμού των κλειδιών που εμπλέκονται στις διαδικασίες της κρυπτογράφησης και αποκρυπτογράφησης. Πιο συγκεκριμένα [4]:

- **Συμμετρικοί αλγόριθμοι ή αλγόριθμοι ιδιωτικού κλειδιού (symmetric ή secret key)**- χρησιμοποιούν ένα μόνο κλειδί για την κρυπτογράφηση και αποκρυπτογράφηση δεδομένων. Η κύρια πρόκληση εδώ είναι η ασφάλεια της μεταφοράς του κλειδιού, καθώς αν κάποιος αποκτήσει πρόσβαση σε αυτό, μπορεί να αποκρυπτογραφήσει όλα τα δεδομένα. Συνηθισμένο παράδειγμα συμμετρικού αλγορίθμου είναι ο AES (Advanced Encryption Standard).
- **Ασύμμετροι αλγόριθμοι ή αλγόριθμοι δημόσιου κλειδιού (asymmetric ή public key)**- χρησιμοποιούν ένα ζεύγος κλειδιών: ένα δημόσιο και ένα ιδιωτικό. Το δημόσιο κλειδί χρησιμοποιείται για την κρυπτογράφηση, ενώ το ιδιωτικό χρησιμοποιείται για την αποκρυπτογράφηση. Αυτό επιτρέπει τη δημιουργία ψηφιακών υπογραφών και ασύμμετρης κρυπτογραφίας, προσφέροντας υψηλό επίπεδο ασφαλείας. Ένα διάσημο παράδειγμα ασύμμετρου αλγορίθμου είναι το RSA.
- **Μονόδρομες συναρτήσεις σύνοψης (one-way hash functions)**- χρησιμοποιούνται για τη δημιουργία ενός σταθερού μεγέθους εξόδου από οποιοδήποτε μέγεθος εισόδου. Είναι σημαντικές για τον έλεγχο ακεραιότητας και τη δημιουργία αναγνωριστικών. Η κυριότερη ιδιότητά τους είναι ότι είναι δύσκολο να αντιστραφούν, προσφέροντας ασφάλεια στη διατήρηση του ακέραιου χαρακτήρα των δεδομένων. Παραδείγματα one-way hash functions περιλαμβάνουν το SHA-256 και το MD5 (αν και το MD5 θεωρείται πλέον ασφαλές για συγκεκριμένες εφαρμογές).



Εικόνα 7: *Three Types of Cryptography [4]*

2.5.1 Συμμετρική Κρυπτογραφία

Στην κρυπτογραφία με μυστικό κλειδί, χρησιμοποιείται ένα μόνο κλειδί τόσο για τη διαδικασία της κρυπτογράφησης όσο και για τη διαδικασία της αποκρυπτογράφησης. Στην **Εικόνα 7Α**, ο αποστολέας χρησιμοποιεί το κλειδί ή ένα συγκεκριμένο σύνολο κανόνων για να κρυπτογραφήσει το απλό κείμενο και στέλνει το κρυπτογραφημένο κείμενο στον παραλήπτη. Ο παραλήπτης χρησιμοποιεί το ίδιο κλειδί ή σύνολο κανόνων για την αποκρυπτογράφηση του μηνύματος και την ανάκτηση του απλού κειμένου. Επειδή το ίδιο κλειδί χρησιμοποιείται για και τις δύο λειτουργίες, αυτή η μέθοδος κρυπτογραφίας είναι επίσης γνωστή ως συμμετρική κρυπτογράφηση.

Σε αυτή τη μορφή κρυπτογραφίας, είναι προφανές ότι το κλειδί πρέπει να είναι γνωστό τόσο στον αποστολέα όσο και στον παραλήπτη. Αυτό, ουσιαστικά, αποτελεί το μυστικό της διαδικασίας. Η κύρια πρόκληση σε αυτήν την προσέγγιση, φυσικά, είναι η ασφαλής διανομή του κλειδιού.

Τα συστήματα κρυπτογραφίας μυστικού κλειδιού κατηγοριοποιούνται συνήθως ως κρυπτογραφήματα ροής (stream ciphers) ή ως κρυπτογραφήματα μπλοκ (block ciphers). Τα stream ciphers λειτουργούν με ένα μόνο bit (byte ή word) κάθε φορά, χρησιμοποιώντας κάποιον μηχανισμό ανατροφοδότησης προκειμένου να μεταβάλλεται διαρκώς το κλειδί (keystream). Από την άλλη, ένα block cipher κρυπτογραφεί ένα μπλοκ δεδομένων κάθε φορά, χρησιμοποιώντας το ίδιο κλειδί για κάθε μπλοκ. Στο block cipher, το ίδιο απλό κείμενο θα μετατρέπεται πάντα στο ίδιο κρυπτοκείμενο (ciphertext) όταν

χρησιμοποιείται το ίδιο κλειδί, ενώ στο stream cipher, το ίδιο απλό κείμενο θα οδηγεί σε διαφορετικό κρυπτοκείμενο.

Τα stream ciphers εντοπίζονται σε διάφορες εκδοχές, αλλά δύο από αυτές είναι ιδιαίτερα σημαντικές. Τα αυτοσυγχρονιζόμενα stream ciphers υπολογίζουν κάθε bit στο keystream ως συνάρτηση των προηγούμενων n bits στο ίδιο keystream. Ο όρος "αυτοσυγχρονιζόμενα" προέρχεται από το γεγονός ότι η διαδικασία αποκρυπτογράφησης μπορεί να παραμείνει συγχρονισμένη με τη διαδικασία κρυπτογράφησης απλώς γνωρίζοντας πόσο μακριά βρίσκεται στο keystream των n bit. Ένα πρόβλημα είναι η διάδοση σφαλμάτων- ένα αλλοιωμένο bit κατά τη μετάδοση θα οδηγήσει σε n αλλοιωμένα bits στην πλευρά λήψης.

Τα σύγχρονα stream ciphers, από την άλλη, παράγουν το keystream ανεξάρτητα από τη ροή του μηνύματος, αλλά χρησιμοποιούν την ίδια συνάρτηση δημιουργίας keystream τόσο στον αποστολέα όσο και στον παραλήπτη. Παρόλο που τα stream ciphers δεν προκαλούν σφάλματα μετάδοσης, λόγω της φύσης τους είναι περιοδικά με αποτέλεσμα το keystream τελικά να επαναλαμβάνεται.

Το βασικό πλεονέκτημα των αλγορίθμων συμμετρικού κλειδιού είναι ότι η διαδικασία της κρυπτογράφησης και αποκρυπτογράφησης είναι αρκετά γρήγορη και δεν απαιτεί σημαντική υπολογιστική ισχύ. Οι πιο γνωστοί αλγόριθμοι αυτού του είδους αποτελούν ο **DES**, **Triple DES**, **IDEA**, **RC2**, **RC4**, και τον **AES**.

Τα στάδια της επικοινωνίας σε ένα συμμετρικό κρυπτοσύστημα είναι τα ακόλουθα:

1. Ο Bob και η Alice αποφασίζουν για ένα κλειδί, το οποίο επιλέγεται τυχαία από τον κλειδοχώρο.
2. Η Alice αποστέλλει το κλειδί στον Bob μέσω ενός ασφαλούς καναλιού.
3. Ο Bob δημιουργεί ένα μήνυμα.
4. Κρυπτογραφεί το μήνυμα με το κλειδί που έλαβε από την Alice, και η παραγόμενη κρυπτοσυμβολοσειρά αποστέλλεται.
5. Η Alice λαμβάνει την κρυπτοσυμβολοσειρά και, στη συνέχεια, με το ίδιο κλειδί, αποκρυπτογραφεί τη σειρά, παράγοντας το αρχικό μήνυμα.

2.5.2 Ασύμμετρη Κρυπτογραφία

Η Κρυπτογραφία δημόσιου κλειδιού (Public Key Cryptography- PKC) θεωρείται μία από τις σημαντικότερες εξελίξεις τα τελευταία 300- 400 χρόνια. Το PKC περιγράφηκε για πρώτη φορά δημόσια από τον καθηγητή Martin Hellman και τον μεταπτυχιακό του φοιτητή Whitfield Diffie του Stanford Πανεπιστημίου το 1976. Η εργασία τους αναφερόταν σε ένα σύστημα κρυπτογράφησης δύο κλειδιών,

στο οποίο δύο μέρη θα μπορούσαν να επικοινωνήσουν με ασφάλεια μέσω ενός μη ασφαλούς καναλιού χωρίς να χρειάζεται να μοιραστούν κάποιο μυστικό κλειδί.

Το PKC εξαρτάται από την ύπαρξη των λεγόμενων μονόδρομων συναρτήσεων ή διαφορετικά μαθηματικών συναρτήσεων που είναι εύκολο να υπολογιστούν, ωστόσο η αντίστροφη συνάρτηση τους είναι σχετικά δύσκολο να υπολογιστεί. Δύο τέτοια παραδείγματα είναι τα εξής [4]:

1. **Πολλαπλασιασμός έναντι παραγοντοποίησης**- Έστω οι δύο αριθμοί 9 και 16 και ζητείται να υπολογιστεί το γινόμενό τους. Όπως είναι προφανές, χρειάζεται ελάχιστος χρόνος για να απαντήσει κάποιος πως το αποτέλεσμα ισούται με 144. Ωστόσο, υποθέτουμε ότι δίνεται ο αριθμός 144 και ζητείται το ζεύγος ακεραίων που πολλαπλασιάστηκαν για να προκύψει αυτός ο αριθμός. Η λύση τελικά θα βρεθεί αλλά ο υπολογισμός του γινομένου χρειάστηκε ελάχιστο χρόνο σε σύγκριση με την παραγοντοποίηση που απαιτείται (πρέπει πρώτα να βρεθεί το ζεύγος των 8 ακεραίων παραγόντων κ.ο.κ.).
2. **Εκθετικά έναντι λογαρίθμων**- Έστω ο αριθμός 3 και ζητείται ο υπολογισμός του 3^6. Είναι εύκολο να υπολογιστεί το αποτέλεσμα (= 729). Ωστόσο, υποθέτουμε ότι δίνεται ο αριθμός 729 και ζητούνται οι δύο αριθμοί, x και y, που χρησιμοποιήθηκαν έτσι ώστε $\log_x 729 = y$. Θα χρειαστεί σίγουρα περισσότερος χρόνος.

Τα παραπάνω παραδείγματα, αν και αρκετά εύκολα, αντιπροσωπεύουν δύο αρκετά λειτουργικά ζεύγη, όπως είναι ο πολλαπλασιασμός και ο υπολογισμός εκθέτη σε σύγκριση με τη σχετική δυσκολία της παραγοντοποίησης και του υπολογισμού λογαρίθμων αντίστοιχα. Αυτό ακριβώς όμως αποτελεί το σημαντικό σημείο στο PKC, δηλαδή η αναζήτηση μιας μονόδρομης συνάρτησης, έτσι ώστε ο αντίστροφος υπολογισμός να γίνεται εύκολα δεδομένης της γνώσης κάποιων επιπρόσθετων πληροφοριών.

Πιο συγκεκριμένα, στο PKC χρησιμοποιούνται δύο κλειδιά που σχετιζόνται μαθηματικά, αν και η γνώση του ενός κλειδιού δεν επιτρέπει σε κάποιον άλλο να προσδιορίσει εύκολα το άλλο κλειδί. Το ένα κλειδί χρησιμοποιείται για την κρυπτογράφηση του plaintext και το άλλο για την αποκρυπτογράφηση του ciphertext. Αυτό που πρέπει να υπογραμμιστεί, είναι ότι δεν έχει σημασία ποιο κλειδί εφαρμόζεται πρώτο αλλά ότι απαιτούνται και τα δύο κλειδιά για να λειτουργήσει η διαδικασία (βλ. **Εικόνα 7Β**). Το γεγονός ότι απαιτείται ένα ζεύγος κλειδιών έχει ως αποτέλεσμα η συγκεκριμένη προσέγγιση να ονομάζεται επίσης ασύμμετρη κρυπτογραφία.

Τα στάδια επικοινωνίας σε ένα ασύμμετρο κρυπτοσύστημα είναι τα ακόλουθα:

1. Η γεννήτρια κλειδιών του Bob παράγει 2 ζεύγη κλειδιών.
2. Η γεννήτρια κλειδιών της Alice παράγει 2 ζεύγη κλειδιών.

3. Η Alice και ο Bob ανταλλάσουν τα δημόσια ζεύγη.
4. Ο Bob παράγει ένα μήνυμα.
5. Κρυπτογραφεί το μήνυμα με το δημόσιο κλειδί της Alice και η παραγόμενη κρυπτοσυμβολοσειρά αποστέλλεται.
6. Η Alice λαμβάνει την κρυπτοσυμβολοσειρά και στην συνέχεια με το ιδιωτικό της κλειδί την αποκρυπτογραφεί και η έξοδος που παράγεται, είναι το μήνυμα.

2.5.2.1 Ο αλγόριθμος RSA

Ο RSA [5] αποτελεί τον πρώτο ολοκληρωμένο αλγόριθμο δημόσιου κλειδιού που αξιοποιείται στην κρυπτογράφηση και στις ψηφιακές υπογραφές. Από όλους τους αλγόριθμους που προτάθηκαν όλα αυτά τα χρόνια, ο RSA είναι μακράν ο πιο εύκολος στην κατανόηση και στην υλοποίηση καθώς και ο πιο δημοφιλής. Πήρε το όνομα του από τους τρεις εφευρέτες του, τον Ron Rivest, τον Adi Shamir και τον Leonard Adleman.

Η ασφάλεια του RSA βασίζεται στη δυσκολία της παραγοντοποίησης μεγάλων αριθμών. Τα δημόσια και ιδιωτικά κλειδιά που χρησιμοποιούνται, είναι συνήθως συναρτήσεις ενός ζεύγους μεγάλων (από 100 έως 200 ψηφία) πρώτων αριθμών. Η ανάκτηση του plaintext μέσω του δημόσιου κλειδιού και το ciphertext θεωρείται ότι είναι ισοδύναμα με την παραγοντοποίηση του γινομένου των δύο πρώτων αριθμών.

Για τη δημιουργία δύο κλειδιών, επιλέγονται δύο τυχαίοι μεγάλοι πρώτοι αριθμοί, p και q. Για μέγιστη ασφάλεια επιλέγονται p και q ίσου μήκους. Έπειτα, υπολογίζεται το γινόμενο

$$n = p * q$$

Στη συνέχεια, γίνεται επιλογή του κλειδιού κρυπτογράφησης, e, προκειμένου το e και το (p- 1) * (q- 1) να είναι σχετικά πρώτοι (relatively prime). Τέλος, χρησιμοποιείται ο εκτεταμένος ευκλείδειος αλγόριθμος [5], για να προκύψει το κλειδί κρυπτογράφησης, d, ώστε

$$ed \equiv 1 \bmod (p- 1)(q- 1)$$

Με άλλα λόγια

$$d = e^{-1} \bmod ((p- 1)(q- 1))$$

Επισημαίνεται ότι τα d και n είναι επίσης σχετικά πρώτοι. Οι αριθμοί e και n αποτελούν το δημόσιο

κλειδί- ο αριθμός d είναι το ιδιωτικό κλειδί. Οι δύο πρώτοι αριθμοί, p και q, δεν χρειάζονται πλέον. Θα πρέπει να απορριφθούν, αλλά να μην αποκαλυφθούν ποτέ.

Για την κρυπτογράφηση ενός μηνύματος m, το τελευταίο διαιρείται πρώτα σε n αριθμητικά μπλοκ- για δυαδικά δεδομένα, προτιμάται η επιλογή της μεγαλύτερης δύναμης του 2 που είναι όμως μικρότερη του n. Πιο συγκεκριμένα, εάν και οι δύο p και q είναι 100ψήφιοι πρώτοι αριθμοί. τότε το n θα έχει κάτι λιγότερο από 200 ψηφία και κάθε μπλοκ μηνύματος, m_i, θα πρέπει να έχει μήκος μικρότερο των 200 ψηφίων. Στην περίπτωση που χρειάζεται να κρυπτογραφηθεί ένας σταθερός αριθμός από μπλοκς, τα τελευταία μπορούν να συμπληρωθούν με μερικά μηδενικά στα αριστερά για να διασφαλιστεί ότι θα είναι πάντα μικρότερα του n. Το κρυπτογραφημένο μήνυμα, c, θα αποτελείται από μπλοκ μηνυμάτων παρόμοιου μεγέθους c_i, περίπου του ίδιου μήκους. Ο τύπος κρυπτογράφησης είναι

$$c_i = m_i^e \bmod n$$

Για την αποκρυπτογράφηση ενός μηνύματος, επιλέγεται κάθε κρυπτογραφημένο μπλοκ c_i και υπολογίζεται το εξής:

$$m_i = c_i^d \bmod n$$

Για την καλύτερη κατανόηση του RSA αλγορίθμου, παρακάτω παρουσιάζεται ένα παράδειγμα.

Έστω p = 47 και q = 71, τότε

$$n = p * q = 3337$$

Το κλειδί κρυπτογράφησης, e, δεν πρέπει να έχει κοινούς παράγοντες με το

$$(p- 1) * (q- 1) = 46 * 70 = 3220$$

Επιλεξτε το e (τυχαία) να είναι το 79. Σε αυτή την περίπτωση

$$d = 79^{-1} \bmod 3220 = 1019$$

Δημοσιεύστε τα e και n και κρατήστε το d μυστικό. Απορρίψτε τα p και q.

Για να κρυπτογραφήσετε το μήνυμα m = 6882326879666683, πρώτα σπάστε το σε μικρά μπλοκ. Τα τριψήφια μπλοκ είναι τα πιο κατάλληλα σε αυτή την περίπτωση. Το μήνυμα χωρίζεται σε έξι μπλοκ, m_i, στα οποία

$$m_1 = 688$$

$$m_2 = 232$$

$$m_3 = 687$$

$$m_4 = 966$$

$$m_5 = 668$$

$$m_6 = 003$$

Το πρώτο μπλοκ κρυπτογραφείται ως εξής:

$$688^{79} \bmod 3337 = 1570 = c_1$$

Η εκτέλεση της ίδιας πράξης στα επόμενα μπλοκ παράγει ένα κρυπτογραφημένο μήνυμα:

$$c = 1570\ 2756\ 2091\ 2276\ 2423\ 158$$

Η αποκρυπτογράφηση του μηνύματος απαιτεί την εκτέλεση του ίδιου πολλαπλασιασμού με τη χρήση του κλειδιού αποκρυπτογράφησης 1019, οπότε

$$1570^{1019} \bmod 3337 = 688 = m_1$$

Το υπόλοιπο μήνυμα μπορεί να ανακτηθεί κατά τον ίδιο τρόπο.

2.5.3 Μονόδρομες Συναρτήσεις Σύνοψης

Οι συναρτήσεις κατακερματισμού, επίσης γνωστές ως αλγόριθμοι κρυπτογράφησης μονής κατεύθυνσης (βλ. **Εικόνα 7C**), αναπτύσσουν αλγόριθμους που, κατά κάποιον τρόπο, παράγουν τιμές κατακερματισμού σταθερού μήκους χωρίς τη χρήση κλειδιού. Αντ' αυτού, υπολογίζεται μια σταθερού μήκους τιμή κατακερματισμού βάσει του plaintext, καθιστώντας αδύνατη την ανάκτηση του περιεχομένου ή του μήκους του. Συχνά, οι αλγόριθμοι κατακερματισμού χρησιμοποιούνται για τη δημιουργία ψηφιακού αποτυπώματος αρχείου, προκειμένου να επαληθευτεί ότι δεν έχει υποστεί τροποποίηση από κάποια εισβολή ή ιό. Επιπλέον, είναι συνηθισμένο να χρησιμοποιούνται για την κρυπτογράφηση κωδικών πρόσβασης από πολλά λειτουργικά συστήματα. Συνεπώς, οι συναρτήσεις κατακερματισμού αποτελούν ένα μέσο για τον έλεγχο της ακεραιότητας ενός αρχείου. Κάποιοι από τους πιο γνωστούς hash αλγορίθμους είναι οι MD (Message Digest), SHA (Secure Hash Algorithm) και Whirlpool [4].

Πιο συγκεκριμένα, οι μονόδρομες συναρτήσεις σύνοψης (one-way hash functions) αποτελούν συναρτήσεις που δέχονται ως είσοδο μια ακολουθία χαρακτήρων μεταβλητού μήκους και παράγουν ένα σταθερού μήκους (συνήθως μικρότερο) μήνυμα, γνωστό ως "τιμή κατακερματισμού" (hash value). Οι μονόδρομες συναρτήσεις σύνοψης δρουν μόνο προς τη μία κατεύθυνση: είναι εύκολο να υπολογιστεί η τιμή κατακερματισμού για ένα δεδομένο μήνυμα, αλλά αδύνατο να ανακτηθεί το μήνυμα από τη συγκεκριμένη τιμή κατακερματισμού. Μια καλά σχεδιασμένη μονόδρομη συνάρτηση σύνοψης θεωρείται ασφαλής αν είναι ανθεκτική στις συγκρούσεις, πράγμα που σημαίνει ότι είναι δύσκολο να βρεθούν δύο μηνύματα που παράγουν την ίδια τιμή κατακερματισμού.

Οι συναρτήσεις κατακερματισμού ενίοτε παρεξηγούνται, ενώ ορισμένες πηγές υποστηρίζουν ότι δύο αρχεία δεν μπορούν να έχουν την ίδια τιμή κατακερματισμού. Αυτή η άποψη βέβαια δεν είναι σωστή. Ας εξετάσουμε μια συνάρτηση κατακερματισμού που παράγει μια τιμή κατακερματισμού 128-bit.

Υπάρχουν, φυσικά, 2^{128} πιθανές τιμές κατακερματισμού. Ωστόσο, υπάρχουν πολύ περισσότερα από 2^{128} πιθανά αρχεία. Επομένως, πρέπει να υπάρχουν πολλά αρχεία- πραγματικά, ένας άπειρος αριθμός- που μπορούν να έχουν την ίδια τιμή κατακερματισμού 128-bit.

Η πρόκληση είναι να εντοπιστούν δύο αρχεία με το ίδιο hash, και αυτό είναι πράγματι εξαιρετικά δύσκολο να επιτευχθεί. Αυτό που είναι πραγματικά δύσκολο βέβαια είναι η δημιουργία ενός αρχείου με συγκεκριμένη τιμή κατακερματισμού, προκειμένου να προκληθεί σύγκρουση. Αυτό είναι το κύριο όφελος των συναρτήσεων κατακερματισμού και ο λόγος που χρησιμοποιούνται εκτενώς σε εφαρμογές ασφάλειας πληροφοριών και εγκληματολογίας υπολογιστών. Δυστυχώς, το 2004, έγινε γνωστό ότι πρακτικές επιθέσεις σύγκρουσης μπορούν να προκληθούν σε αλγορίθμους όπως το MD5, το SHA-1 και άλλους. Αυτό οδήγησε στην ανάγκη αναζήτησης πιο ασφαλών αλγορίθμων κατακερματισμού. Παρόλα αυτά, προς το παρόν, δεν υπάρχει ένας προφανής διάδοχος των MD5 και SHA-1 που μπορεί να χρησιμοποιηθεί γρήγορα. Λόγω της ευρείας χρήσης αυτών των συναρτήσεων κατακερματισμού σε πολλά προϊόντα, η εξάλειψη της χρήσης τους μπορεί να απαιτήσει πολλά χρόνια.

3

Μαθηματικό Υπόβαθρο

Το Galois Field, γνωστό και ως πεδίο Galois, είναι μία μαθηματικό έννοια που προέκυψε από το έργο του Évariste Galois, ενός Γάλλου μαθηματικού του 19ου αιώνα. Τα πεδία Galois έχουν εφαρμογές σε διάφορους τομείς, με την πιο ευρέως γνωστή να είναι στη θεωρία κωδικοποίησης και τους διορθωτικούς κώδικες. Αυτά τα πεδία παρέχουν ένα μαθηματικό πλαίσιο για την εκτέλεση πράξεων όπως πρόσθεση και πολλαπλασιασμός σε ένα πεπερασμένο αριθμό στοιχείων, όπως συμβαίνει στον κόσμο των ψηφιακών επικοινωνιών και των συστημάτων αποθήκευσης δεδομένων. Η θεωρία των πεδίων Galois συνεχίζει να έχει ευρεία σημασία σε πολλούς κλάδους της μαθηματικής και της εφαρμοσμένης επιστήμης.

Τα πεδία Galois, συχνά αναφερόμενα ως πεπερασμένα πεδία ή GF(p), λειτουργούν σε ένα πλαίσιο πεπερασμένου αριθμού στοιχείων, όπου το "p" είναι ένας πρώτος αριθμός. Ένα απλό παράδειγμα είναι το GF(2), που αντιπροσωπεύει το πεδίο Galois με έναν πρώτο αριθμό p=2.

Στο GF(p), οι πράξεις πρόσθεσης και πολλαπλασιασμού ορίζονται με βάση τους κανόνες της πράξης modulo p. Αυτό σημαίνει ότι ανάγουμε το αποτέλεσμα κάθε πράξης σε όλους τους ακέραιους που βρίσκονται στο διάστημα [0, p-1].

Για παράδειγμα, σε ένα GF(2), ο πολλαπλασιασμός και η πρόσθεση είναι πολύ απλοποιημένοι. Αν προσθέσουμε ή πολλαπλασιάσουμε δύο αριθμούς στο GF(2), το αποτέλεσμα θα είναι είτε 0 είτε 1, καθώς η πρόσθεση και ο πολλαπλασιασμός modulo 2 είναι τα ίδια.

3.1 Galois Field

Τα στοιχεία του πεδίου Galois ορίζονται ως

$$gf(p^n) = (0, 1, 2, .., p - 1) \cup$$
$$(p, p + 1, p + 2, .., p + p - 1) \cup$$
$$(p\ 2, p2 + 1, p2 + 2, .., p2 + p - 1) \cup .. \cup$$
$$(p\ n-1, pn-1 + 1, pn-1 + 2, .., pn-1 + p - 1)$$

όπου $p \in P$ and $n \in Z^+$. Η τάξη του πεδίου ορίζεται από τη σχέση p^n, όπου το p ονομάζεται χαρακτηριστικό του πεδίου. Αντίστοιχα, ο όρος "gf" αναφέρεται στο πεδίο Galois. Είναι σημαντικό να σημειώσουμε ότι ο βαθμός του πολυωνύμου κάθε στοιχείου του πεδίου είναι το πολύ n- 1.

Παράδειγμα το

$$gf(5) = (0, 1, 2, 3, 4)$$

αποτελείται από 5 στοιχεία όπου καθένα από αυτά είναι πολυώνυμο βαθμού 0(μια σταθερά) ενώ

$$gf(2^3) = (0, 1, 2, 2 + 1, 2^2, 2^2 + 1, 2^2 + 2, 2^2 + 2 + 1)$$
$$= (0, 1, 2, 3, 4, 5, 6, 7)$$

το οποίο αποτελείται από $2^3 = 8$ στοιχεία, όπου καθένα από αυτά είναι ένα πολυώνυμο βαθμού το πολύ 2.

3.2 Δυαδικό Σύστημα

Στο δυαδικό αριθμητικό σύστημα ή σύστημα βάσης 2, κάθε τιμή αναπαρίσταται με 0 και 1. Για να μεταφερθεί το δεκαδικό αριθμητικό σύστημα ή το σύστημα βάσης 10 σε δυαδικό σύστημα, πρέπει ο δεκαδικός αριθμός να αναπαρασταθεί ως άθροισμα $a_n 2^n$. Δηλαδή, αν x είναι ο εν λόγω δεκαδικός, τότε ισχύει

$$x = \sum_{n \in \mathbb{N}} a_n 2^n$$

Οι συντελεστές a_n γράφονται στη συνέχεια κατά φθίνουσα σειρά του n και όλα τα μηδενικά που προηγούνται, παραλείπονται. Το τελικό αποτέλεσμα είναι η δυαδική αναπαράσταση του δεκαδικού x. Τελικά, το δυαδικό σύστημα προσφέρει έναν εναλλακτικό τρόπο αναπαράστασης των στοιχείων ενός πεδίου Galois. Τόσο η πολυωνυμική όσο και η δυαδική αναπαράσταση ενός στοιχείου έχει τα δικά της πλεονεκτήματα και μειονεκτήματα.

Παράδειγμα το

$$19 =. .. + 1 \cdot 2^4 + 0 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0$$

οπότε η δυαδική αναπαράσταση του 19 είναι 10011, ενώ τα στοιχεία του $gf(2^3)$ σε δυαδική μορφή είναι

$$gf(2^3) = (001, 010, 011, 100, 101, 110, 111)$$

3.3 Bit και Byte

Κάθε ψηφίο, είτε 0 είτε 1, ονομάζεται bit. Ένα bit ανήκει στο πεδίο Galois τάξης 2, καθώς μπορεί να εκφραστεί ως στοιχείο του gf(2). Επιπλέον, υπάρχει το byte που αποτελείται από 8 bits, και επομένως ανήκει στο πεδίο Galois τάξης 2 και 2^8 $gf(2^8)$. Στον τομέα της κρυπτογραφίας υπολογιστών, όπου τα δεδομένα παρουσιάζονται ως σειρές από bytes, είναι σημαντικό να επικεντρωθούμε στο πεδίο Galois τάξης 2 και 2^8, που αποτελεί τον χώρο στον οποίο εκτελούνται οι περισσότερες λειτουργίες.

Επειδή ο υπολογιστής αποθηκεύει τα δεδομένα σε bytes, κάθε δυαδικός αριθμός πρέπει να έχει μήκος 8 bit. Για αριθμούς με μήκος μικρότερο των 8 bit, προστίθενται μηδενικά στην αρχή. Έτσι, ο μεγαλύτερος αριθμός που μπορεί να αποθηκεύσει ένα byte είναι 11111111, που αναπαρίσταται ως $2^7 + 2^6 + 2^5 + 2^4 + 2^3 + 2^2 + 2^1 + 2^0$ και ισούται με 255. Από τα προηγούμενα προκύπτει το παράδειγμα όπου το 19 αποθηκεύεται ως 00010011 σε μορφή byte.

3.4 Αριθμητική Πεπερασμένου Πεδίου

Σε αντίθεση με τον ευκλείδειο χώρο, η πρόσθεση (και η αφαίρεση) και ο πολλαπλασιασμός στο πεδίο Galois απαιτούν πρόσθετα βήματα.

3.4.1 Πρόσθεση και Αφαίρεση

Μια πρόσθεση στο πεδίο Galois είναι αρκετά απλή. Ας υποθέσουμε ότι τα f(p) και g(p) είναι πολυώνυμα στο $gf(p^n)$. Έστω $A = a_{n-1}a_{n-2} \dots a_1a_0$, $B = b_{n-1}b_{n-2} \dots b_1b_0$ και $C = c_{n-1}c_{n-2} \dots c_1c_0$ οι συντελεστές των f(p), g(p) και h(p) = f(p) + g(p) αντίστοιχα. Αν a_k, b_k και c_k είναι οι συντελεστές του p^k στους f(p), g(p), και h(p) αντίστοιχα, τότε

$$c_k = a_k + b_k \pmod p$$

Ομοίως, αν h(p) = f(p)- g(p) τότε

$$c_k = a_k - b_k \pmod p$$

όπου $0 \leq k \leq n-1$. Δεδομένου ότι ο υπολογιστής λειτουργεί στο $gf(2^8)$, αν τα a_k και b_k αναφέρονται στο k-οστό bit στα bytes που πρέπει να γίνει η πρόσθεση, τότε το c_k, που αντιπροσωπεύει το k-οστό bit στο προκύπτον byte, δίνεται από τη σχέση

$$c_k = a_k + b_k \pmod 2$$

Δεδομένου ότι $0 + 1 = 1 + 0 = 1 \pmod 2 = 1$ και $0 + 0 = 0 \pmod 2 = 1 + 1 = 2 \pmod 2 = 0$, η πρόσθεση θεωρείται πράξη αποκλειστικού ή που είναι επίσης γνωστή και ως πράξη XOR. Δηλαδή, η πράξη XOR επιστρέφει 0 εάν και οι δύο καταχωρήσεις είναι ίσες και επιστρέφει 1 σε αντίθετη περίπτωση, πράγμα που σημαίνει επίσης ότι η αφαίρεση και η πρόσθεση είναι ίδιες στο πεδίο Galois του οποίου το χαρακτηριστικό πεδίο είναι το 2. Λόγω της φύσης του πεδίου Galois, η πρόσθεση και η αφαίρεση δύο bytes δεν θα υπερβαίνει το 11111111 = 255, τη μεγαλύτερη τιμή που μπορεί να αποθηκεύσει ένα byte, και επομένως είναι μια ασφαλής πράξη.

Παράδειγμα

Ας υποθέσουμε ότι εργαζόμαστε στο $gf(2^8)$, τότε $83 + 249$ είναι

$$83 + 249 = (2^6 + 2^4 + 2^1 + 2^0) + (2^7 + 2^6 + 2^5 + 2^4 + 2^3)$$

$$= 2^7 + 2 \cdot 2^6 + 2^5 + 2 \cdot 2^4 + 2^3 + 2^1 + 2 \cdot 2^0$$

$$= 2^7 + 2^5 + 2^3 + 2^1 = 169$$

Εναλλακτικά, όσον αφορά στο δυαδικό αριθμητικό σύστημα

$$83 + 249 = 01010011 + 11111001$$

$$= 10101010 = 169$$

και τα αποτελέσματα συμπίπτουν.

3.4.2 Πολλαπλασιασμός και Πολλαπλασιαστικό Αντίστροφο

Ο πολλαπλασιασμός στο πεδίο Galois απαιτεί πιο περίπλοκη εργασία. Ας υποθέσουμε ότι τα πολυώνυμα f(p) και g(p) ανήκουν στο $gf(p^n)$, και έστω m(p) ένα μη αναγώγιμο πολυώνυμο (ή ένα πολυώνυμο που δεν μπορεί να παραγοντοποιηθεί) βαθμού τουλάχιστον n στο $gf(p^n)$. Θέλουμε το m(p) να είναι ένα πολυώνυμο βαθμού τουλάχιστον n, ώστε το γινόμενο των f(p) και g(p) να μην υπερβαίνει το 11111111 (255), καθώς το γινόμενο πρέπει να αποθηκευτεί ως ένα byte. Αν h(p) αποτελεί το προκύπτον προϊόν, τότε

$$h(p) = (f(p) \cdot g(p)) \pmod{m(p)}$$

Από την άλλη πλευρά, το πολλαπλασιαστικό αντίστροφο της f(p) δίνεται από την a(p), με τέτοιο τρόπο ώστε

$$(f(p) \cdot a(p)) \pmod{m(p)} = 1$$

Σημειώνεται ότι ο υπολογισμός του γινομένου δύο πολυωνύμων και του πολλαπλασιαστικού αντίστροφου ενός πολυωνύμου απαιτεί τόσο τη μείωση των συντελεστών modulo p όσο και τη μείωση των πολυωνύμων modulo m(p). Το μειωμένο πολυώνυμο μπορεί να υπολογιστεί εύκολα με μεγάλη διαίρεση, ενώ ο καλύτερος τρόπος για τον υπολογισμό του αντίστροφου πολλαπλασιαστικού είναι με τη χρήση του εκτεταμένου ευκλείδειου αλγορίθμου [7]. Οι λεπτομέρειες για τους υπολογισμούς στο $gf(2^8)$ εξηγούνται καλύτερα στα ακόλουθα παραδείγματα.

Παράδειγμα 1

Ας υποθέσουμε ότι εργαζόμαστε στο $gf(2^8)$ και παίρνουμε το μη αναγώγιμο πολυώνυμο modulo m(p) να είναι $p^8 + p^6 + p^5 + p^1 + p^0$. Για να υπολογίσουμε το $84 \cdot 13$, πρέπει να ακολουθηθούν διάφορα βήματα. Πρώτον, υπολογίζουμε το γινόμενο του πολυωνύμου και μειώνουμε τους συντελεστές modulo 2.

$$84 \cdot 13 = ((2^6 + 2^4 + 2^2) \cdot (2^3 + 2^2 + 2^0)) \pmod{m(p)}$$

$$= (2^9 + 2^8 + 2^7 + 2 \cdot 2^6 + 2^5 + 2 \cdot 2^4 + 2^2) \pmod{m(p)}$$

$$= (2^9 + 2^8 + 2^7 + 2^5 + 2^2) \pmod{m(p)}$$

Στη συνέχεια χρησιμοποιούμε τη μεγάλη διαίρεση για να υπολογίσουμε το μειωμένο πολυώνυμο ως εξής

Remainder	Quotient
$2^9 + 2^8 + 2^7 + 2^5 + 2^2$	
$2^8 + 2^6 + 2^5 + 2^1 + 2^0$	
2^0	$2^1 + 2^0$

όπου η τελευταία εγγραφή στην πρώτη στήλη είναι το προϊόν που αναζητούμε ($= 2^0$). Εφόσον το γινόμενο είναι 1, προκύπτει ότι το 84 και 13 είναι πολλαπλασιαστικά αντίστροφα ζεύγη.

Παράδειγμα 2

Σε αυτό το παράδειγμα δεν γνωρίζουμε το πολλαπλασιαστικό αντίστροφο του 84. Αυτό σημαίνει πως για να υπολογίσουμε το πολλαπλασιαστικό αντίστροφο, θα χρησιμοποιήσουμε τον Εκτεταμένο Ευκλείδειο Αλγόριθμο. Σε αντίθεση με τη μακρά διαίρεση, πρέπει να παρακολουθούμε το βοηθητικό (auxiliary) όταν δουλεύουμε με τον εκτεταμένο ευκλείδειο αλγόριθμο ως εξής:

Remainder	Quotient	Auxiliary
$2^8 + 2^6 + 2^5 + 2^1 + 2^0$		0
$2^6 + 2^4 + 2^2$		1
$2^5 + 2^4 + 2^1 + 2^0$	2^2	2^2
2^0	$2^1 + 2^0$	$2^3 + 2^2 + 1$

Οι δύο πρώτες σειρές στη στήλη Remainder είναι πάντα το πολυώνυμο modulo ακολουθούμενο από το πολυώνυμο που θέλουμε να αντιστρέψουμε. Οι δύο πρώτες γραμμές στη στήλη στη στήλη Remainder είναι πάντα το πολυώνυμο modulo ακολουθούμενο από το πολυώνυμο που θέλουμε να αντιστρέψουμε. Οι δύο πρώτες σειρές στη στήλη Auxiliary είναι πάντα 0 και 2^0. Το υπόλοιπο και το πηλίκο στη σειρά n υπολογίζεται από τη διαίρεση των υπολοίπων στη σειρά n- 1 και n- 2, ενώ το auxiliary στη σειρά n δίνεται από το άθροισμα των βοηθητικών στη σειρά n- 2 και του γινομένου του πηλίκου και του βοηθητικού στη σειρά n- 1 μέχρι το τελευταίο υπόλοιπο να ισούται με 2^0. Η τελευταία εγγραφή στο βοηθητικό τυχαίνει να είναι το πολλαπλασιαστικό αντίστροφο του γινομένου, επομένως το πολλαπλασιαστικό αντίστροφο του 84 είναι 23 + 22 + 20 = 13 που συμφωνεί με το προηγούμενο παράδειγμα.

3.5 Ανάγκη Χρήσης Πεδίων GF(2n) στην Κρυπτογραφία

Σχεδόν όλοι οι αλγόριθμοι κρυπτογράφησης, είτε πρόκειται για συμμετρικούς είτε για κοινού κλειδιού, περιλαμβάνουν αριθμητικές πράξεις μεταξύ ακεραίων. Εάν μια από αυτές τις πράξεις είναι η διαίρεση, τότε είναι αναγκαία η χρήση πεπερασμένης αριθμητικής σε ένα πεδίο. Για ευκολία και αποτελεσματικότητα υλοποίησης, θα ήταν βολική η ύπαρξη ακεραίων που χωράνε ακριβώς σε έναν δοσμένο αριθμό bits, χωρίς πλεονασμό σε bit patterns. Με βάση αυτό, εργαζόμαστε με ακεραίους από το 0 μέχρι το 2^n- 1, οι οποίοι χωρούν ακριβώς σε ένα n-bit word.

Αν θέλουμε να ορίσουμε έναν αλγόριθμο κρυπτογράφησης που εργάζεται πάνω σε δεδομένα 8 bit κάθε φορά και πρέπει να πραγματοποιήσουμε διαίρεση, με τα 8 bits αναμένουμε αριθμούς μεταξύ 0 και 255. Ωστόσο, το 256 δεν είναι πρώτος αριθμός, και επομένως η αριθμητική δεν μπορεί να εκτελεστεί στο Z_{256} (αριθμητικό modulo 256), άρα δεν αποτελεί πεπερασμένο πεδίο. Ο κοντινότερος πρώτος αριθμός μικρότερος από το 256 είναι το 251. Έτσι, το σύνολο Z_{251}, που χρησιμοποιεί αριθμητική modulo 251, είναι ένα πεπερασμένο πεδίο. Ωστόσο, σε αυτήν την περίπτωση, όταν χρησιμοποιούνται 8-bit patterns, οι αριθμοί 251 μέχρι 255 δεν θα χρησιμοποιούνται, και κατά συνέπεια δεν εκμεταλλεύεται αποτελεσματικά ο χώρος, καθώς δεν εκτελείται αριθμητική σε ολόκληρο το πεδίο.

Όπως παρουσιάστηκε στα προηγούμενα παραδείγματα, αν χρειάζεται να χρησιμοποιηθούν όλες οι αριθμητικές πράξεις, τότε πρέπει να καλυφθεί όλο το εύρος των ακεραίων. Αυτό είναι σημαντικό προκειμένου να αξιοποιηθεί πλήρως το δυναμικό του αριθμητικού πεδίου και να εξασφαλιστεί η αποτελεσματικότητα του αλγορίθμου.

4

Το Πρότυπο Κρυπτογράφησης AES

Σε αυτό το κεφάλαιο, θα εξεταστούν τα βασικά στοιχεία του αλγορίθμου Advanced Encryption Standard (AES).

4.1 Advanced Encryption Standard

Το National Institute of Standards and Technology (NIST) κυκλοφόρησε τον Advanced Encryption Standard (AES) το 2001. Ο αλγόριθμος AES ανήκει στην κατηγορία των συμμετρικών αλγορίθμων κρυπτογραφίας, σύμφωνα με την οποία τόσο ο αποστολέας όσο και ο παραλήπτης συμφωνούν σε ένα κοινό μυστικό κλειδί, προκειμένου να πραγματοποιηθεί η διαδικασία κρυπτογράφησης και αποκρυπτογράφησης. Ο AES λειτουργεί ως συμμετρικός block cipher, ο οποίος χειρίζεται 128-bit block δεδομένα εισόδου και εξόδου ενώ ακόμη κρυπτογραφεί και αποκρυπτογραφεί blocks, χρησιμοποιώντας ένα κλειδί που μπορεί να έχει μέγεθος 256-bit, 192-bit ή 128-bit.

Επιπλέον, ο AES λειτουργεί ως επαναληπτικός αλγόριθμος, με ένα από τα βασικότερα του χαρακτηριστικά να είναι η απλότητα. Το παραπάνω επιτυγχάνεται μέσω της εφαρμογής επαναλαμβανόμενων διαδικασιών αντικατάστασης και αναδιάταξης σε διάφορους γύρους. Πιο συγκεκριμένα, ο AES κρυπτογραφεί ή αποκρυπτογραφεί ένα 128-bit plaintext ή ciphertext εφαρμόζοντας επαναλαμβανόμενα τον ίδιο γύρο μετασχηματισμού αρκετές φορές, ανάλογα με το μέγεθος του κλειδιού.

Στον παρακάτω πίνακα παρουσιάζεται αυτή η εξάρτηση.

	Key length		Block size		Number of rounds
	Nk	**in bits**	***Nb***	**(in bits)**	***Nr***
AES-128	4	128	4	128	10
AES-192	6	192	4	128	12
AES-256	8	256	4	128	14

Πίνακας 1: *Key-Block-Round Combinations*

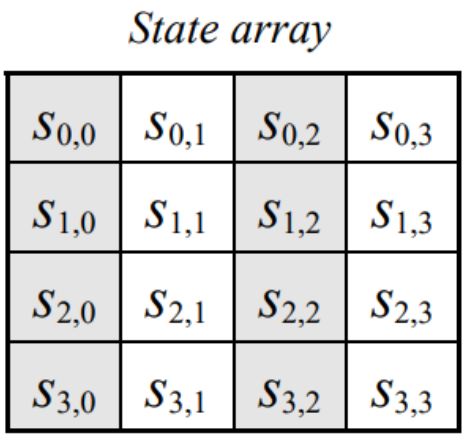
Το block δεδομένων των 128-bit χωρίζεται σε 16 bytes, τα οποία αντιστοιχούν σε έναν 4x4 πίνακα που ονομάζεται State Array. Όλες οι εσωτερικές πράξεις του αλγορίθμου AES εκτελούνται σε αυτόν τον πίνακα. Κάθε byte του State Array αναπαρίσταται ως (i, j) όπου S ($0 \leq i, j \leq 3$), και αποτελεί στοιχείο του πεπερασμένου πεδίου GF(2^8). Γενικά, χρησιμοποιούνται διάφορα ανάγωγα (irreducible) πολυώνυμα για τη δημιουργία του πεπερασμένου πεδίου GF(2^8), ωστόσο το ανάγωγο πολυώνυμο που χρησιμοποιείται στον AES αλγόριθμο είναι το

$$p(x) = x^8 + x^4 + x^3 + x + 1$$

Σύμφωνα με το επιθυμητό επίπεδο ασφαλείας (security level) γίνεται και η επιλογή του πραγματικού μεγέθους του κλειδιού. Ο αλγόριθμος AES-128 είναι ο πλέον διαδεδομένος με αποτέλεσμα να υποστηρίζεται από πολλές υλοποιήσεις. Σε αυτή την ενότητα εξετάζεται κυρίως η λειτουργία κρυπτογράφησης AES με κλειδί μεγέθους 128-bit.

4.1.1 AES Cipher

Όπως φαίνεται και στον παραπάνω πίνακα (βλέπε Πίνακας 1) για κλειδί μεγέθους 128-bit, υπάρχουν 10 γύροι από αντικαταστάσεις (substitutions) και μεταθέσεις (permutations) που πρέπει να εφαρμοστούν στον AES Cipher. Το 128-bit plaintext εισόδου αποθηκεύεται σε έναν πίνακα 4x4 bytes. Κάθε γραμμή και στήλη του πίνακα αποτελείται από 32 bits. Αυτός ο πίνακας, όπως αναφέρθηκε και παραπάνω, ονομάζεται State Array και παρουσιάζεται στην **Εικόνα 8**.



Εικόνα 8: *Illustration of State Array [1]*

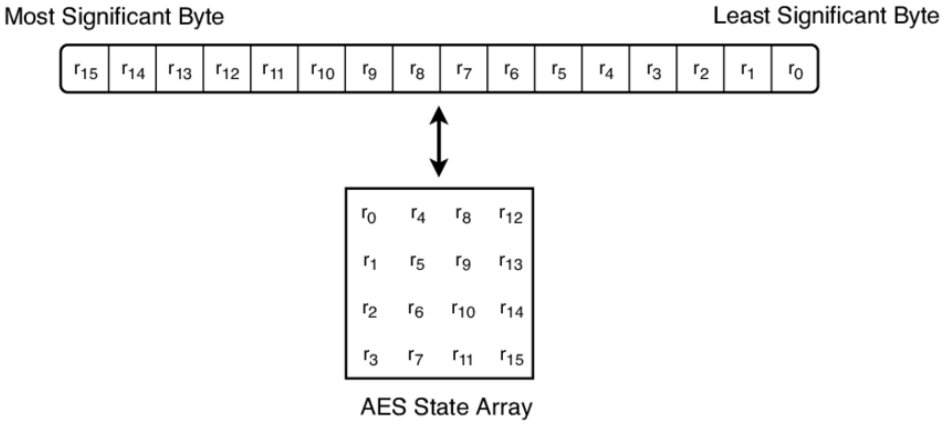
Στην **Εικόνα 8**, το $S_{i,j}$ αναπαριστά ένα byte όπου $0 \leq i, j \leq 3$. Ο State Array είναι ο πίνακας που σε κάθε γύρο μεταβάλλεται. Το κλειδί εισόδου επεκτείνεται και τοποθετείται σε έναν πίνακα των 44 32-bit words ώστε σε κάθε γύρο να αξιοποιούνται τα 4 words (128 bits). Σημειώνεται ότι η επέκταση του κλειδιού πραγματοποιείται πριν από τη λειτουργία του cipher.

Στον αλγόριθμο AES κατά τη διαδικασία της κρυπτογράφησης κάθε γύρος, εκτός από τον τελευταίο, αποτελείται από τέσσερις υποχρεωτικούς μετασχηματισμούς: SubBytes, ShiftRows, MixColumns και AddRoundKey. Ο μετασχηματισμός MixColumns είναι αυτός που δεν συμπεριλαμβάνεται στον τελικό κύκλο. Παρακάτω αναφέρονται συνοπτικά αυτοί οι βασικοί μετασχηματισμοί καθώς και οι ιδιότητές τους.

- **SubBytes**- αποτελεί έναν μη γραμμικό μετασχηματισμό όπου κάθε byte αντικαθίσταται σύμφωνα με έναν πίνακα αντικατάστασης.
- **ShiftRows**- αποτελεί βήμα αλληλομετάθεσης όπου κάθε γραμμή από το State Array ολισθαίνει κυκλικά για έναν συγκεκριμένο αριθμό θέσεων.
- **MixColumns**- εφαρμόζεται σε κάθε στήλη του State Array και αναπαριστά ένα γραμμικό μετασχηματισμό βασισμένο στα 4 bytes της στήλης.
- **AddRoundKey**- κάθε byte του State Array συνδυάζεται με ένα byte από το κλειδί του τρέχοντος κύκλου. Κάθε κλειδί του κύκλου προκύπτει από το κλειδί κρυπτογράφησης χρησιμοποιώντας το σύστημα παραγωγής κλειδιών του AES (key schedule).

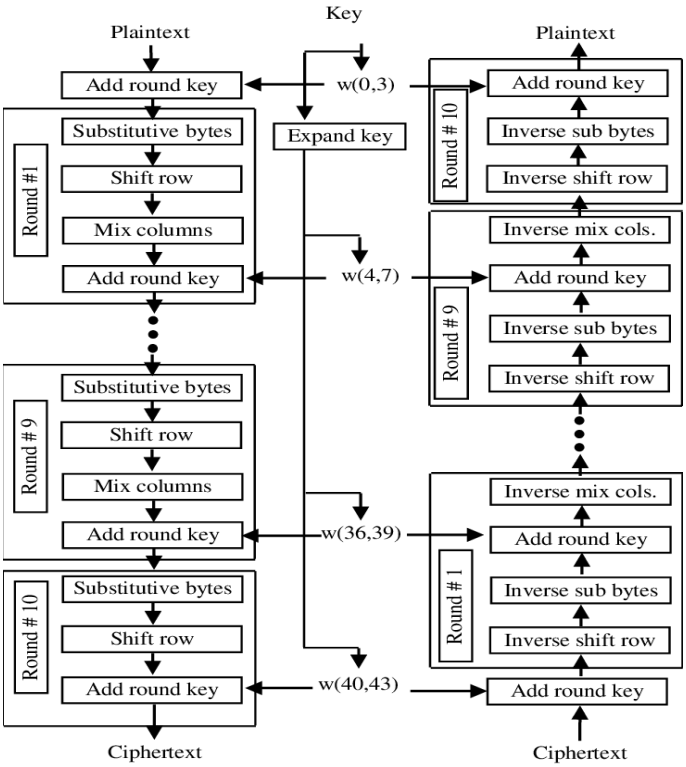
Οι μετασχηματισμοί που αναφέρθηκαν παραπάνω εφαρμόζονται και κατά τη διαδικασία της αποκρυπτογράφησης αλλά με αντίθετη λογική. Πιο συγκεκριμένα, οι μετασχηματισμοί που προκύπτουν είναι οι InvShiftRows, InvSubBytes, InvMixColumns και AddRoundKey.

Στις ενότητες που ακολουθούν, χρησιμοποιείται κατά κύριο λόγο το State Array Block προκειμένου να περιγραφούν τα διάφορα round operations. Επομένως, καθίσταται σημαντική η κατανόηση του τρόπου με τον οποίο η είσοδος που δίνεται στον αλγόριθμο AES μετατρέπεται στον State Array. Στην **Εικόνα 9** παρακάτω παρουσιάζεται αυτή η μετατροπή, κατά την οποία ουσιαστικά κάθε στήλη του State Array γεμίζεται με bytes δεδομένων. Μετά το τέλος της AES κρυπτογράφησης, το State Array που προκύπτει από τον τελικό γύρο, μετατρέπεται πάλι πίσω σε ένα 128-bit stream.



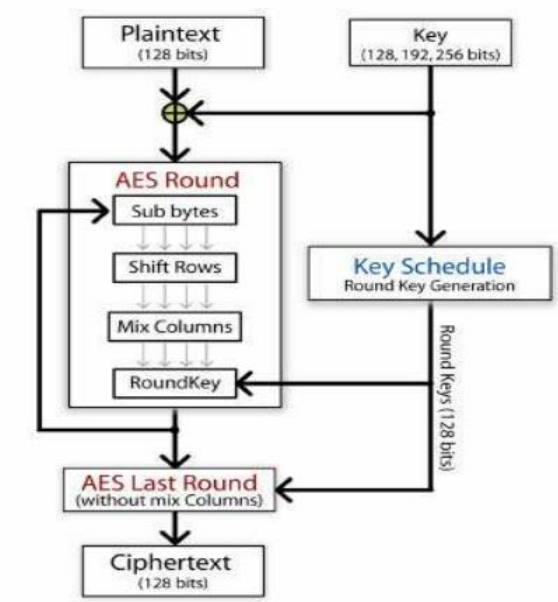
Εικόνα 9: *Mapping between a 128-bit Input and the State Array*

Το διάγραμμα των βασικών στοιχείων του AES αλγορίθμου για την κρυπτογράφηση και την αποκρυπτογράφηση που φαίνεται στην **Εικόνα 10**, υποδεικνύει την σειρά κατά την οποία είναι συνδεδεμένα αυτά τα στοιχεία καθώς και τον αριθμό των γύρων που απαιτούνται σύμφωνα με το μέγεθος του κλειδιού. Στην συνέχεια, θα αναλυθούν οι 4 μετασχηματισμοί που λαμβάνουν χώρα κατά τη διαδικασία της κρυπτογράφησης στον AES.



Εικόνα 10: *AES Functions for Encryption & Decryption*

Στο διάγραμμα που ακολουθεί στην **Εικόνα 11** απεικονίζεται η γενική δομή κάθε κύκλου του AES που επαναλαμβάνεται σύμφωνα με το κλειδί εισόδου.



Εικόνα 11: *AES Round Structure*

4.2 SubBytes Μετασχηματισμός

Ο μετασχηματισμός SubBytes αποτελεί τον μόνο μη γραμμικό μετασχηματισμό που εφαρμόζεται ανεξάρτητα σε κάθε byte του State Array κάνοντας χρήση ενός πίνακα αντικατάστασης. Πιο συγκεκριμένα, όλα τα bytes του State Array αντικαθίστανται σύμφωνα με ένα substitution table, το οποίο ουσιαστικά αποτελεί έναν αντιστρέψιμο πίνακα 16x16 από bytes και συχνά καλείται S-box. Το S-box ορίζεται ως ένα look-up-table (LUT) που χρησιμοποιείται στον SubBytes μετασχηματισμό καθώς περιέχει τα αποτελέσματα των αντικαταστάσεων και μεταθέσεων όλων των πιθανών 8-bit τιμών. Το περιεχόμενο του παραπάνω πίνακα υπολογίζεται ως εξής [1]:

1. Εφαρμογή της αντιστροφής στο πεπερασμένο πεδίο GF(2^8). Το στοιχείο {00} αντιστοιχεί στον εαυτό του.
2. Εφαρμογή του παρακάτω μετασχηματισμού στο GF(2).

$$b'_i = b_i \{xor\} b_{(i+4)} \{xor\} b_{(i+5)mod8} \{xor\} b_{(i+6)mod8} \{xor\} b_{(i+7)mod8} \{xor\} c_i$$

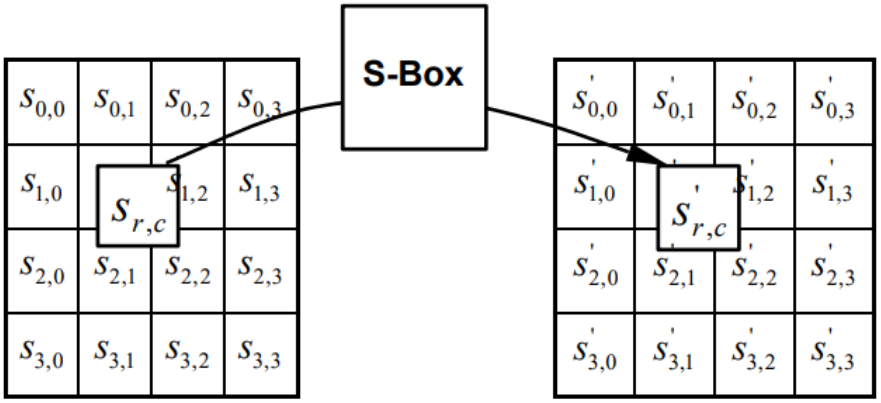
Αυτή η μετασχηματιστική διαδικασία ονομάζεται affine μετασχηματισμός, όπου το b_i αναφέρεται στο i-οστό bit του byte b και το c_i στο i-οστό bit του byte c, με τιμή που είναι είτε {63} είτε {01100011}. Σε μορφή πίνακα, ο δεύτερος μετασχηματισμός του S-box παρουσιάζεται παρακάτω.

$$\begin{bmatrix} b'_0 \\ b'_1 \\ b'_2 \\ b'_3 \\ b'_4 \\ b'_5 \\ b'_6 \\ b'_7 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}.$$

Εικόνα 12: *The Affine Transformation in Register Form*

Αυτό σημαίνει ότι πρώτα εντοπίζει το πολλαπλασιαστικό αντίστροφο σε ένα πεπερασμένο πεδίο (finite-field) και στη συνέχεια εφαρμόζει έναν affine μετασχηματισμό στο GF(2) (ένα μετασχηματισμό που περιλαμβάνει πολλαπλασιασμό με έναν πίνακα ακολουθούμενο από την πρόσθεση ενός διανύσματος). Κάθε byte του State Array αντιστοιχεί σε ένα byte του S-box όπου τα 4 αριστερότερα bits χρησιμοποιούνται ως δείκτες γραμμών (row index) και τα 4 δεξιότερα bits ως δείκτες στηλών (column index). Στην **Εικόνα 13** απεικονίζεται ο τρόπος με τον οποίο ο μετασχηματισμός SubBytes διαμορφώνει

τον State Array. Το S-box έχει σχεδιαστεί με τέτοιο τρόπο ώστε να αντιστέκεται σε κρυπταναλυτικές επιθέσεις [2]. Η σημαντική ιδιότητα που διαθέτει ο SubBytes μετασχηματισμός, είναι ότι η έξοδος του δεν μπορεί να περιγραφεί ως μία απλή μαθηματική συνάρτηση της εισόδου του.



Εικόνα 13: *SubBytes Operation applies the S-box to each Byte of the State Array [1]*

Το S-box που χρησιμοποιείται στον μετασχηματισμό SubBytes και αποτελείται από στοιχεία σε δεκαεξαδική μορφή, εκφράζεται μέσω του πίνακα που παρουσιάζεται στην **Εικόνα 14**, όπως φαίνεται στο επίσημο έγγραφο για τον AES του NIST..

Για παράδειγμα, αν $S_{i,j} = \{53\}$, τότε η τιμή αντικατάστασης θα καθοριστεί από την τομή της γραμμής '5' και της στήλης '3' του **Εικόνα 14**. Ως αποτέλεσμα το '53' θα αντικατασταθεί από την τιμή {ed}.

		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
	1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
	2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
	3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
	4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
	5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
	6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
	7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
	8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
	9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
	a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
	b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
	c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
	d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
	e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
	f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Εικόνα 14: *S-box Substitution Values for the Byte xy (in Hexadecimal Format)*

Σύμφωνα με αυτά που αναλύθηκαν προηγουμένως, στο παρακάτω σχήμα παρουσιάζεται ένα παράδειγμα του μετασχηματισμού SubBytes.

EA	04	65	85
83	45	5D	96
5C	33	98	B0
F0	2D	AD	C5

→

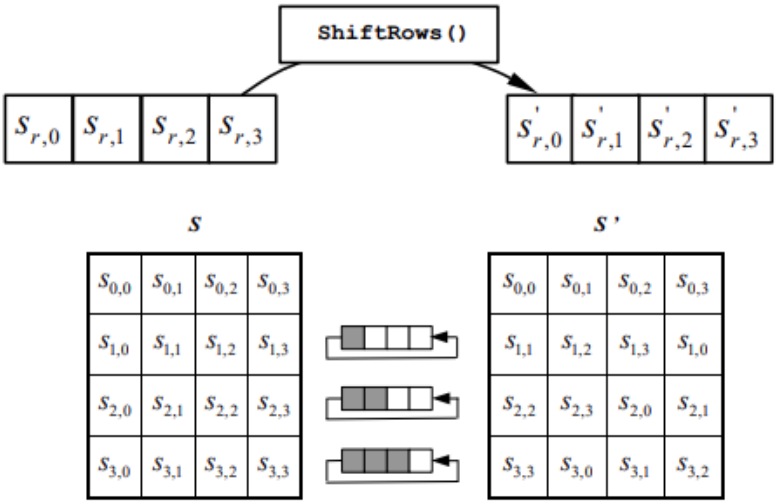
87	F2	4D	97
EC	6E	4C	90
4A	C3	46	E7
8C	D8	95	A6

Σημαντικό να αναφερθεί είναι πως ο πίνακας S-Box τυπικά δεν υπολογίζεται κατά την διαδικασία της κρυπτογραφίας, αλλά οι τιμές του έχουν προϋπολογιστεί.

4.3 ShiftRows Μετασχηματισμός

Στον συγκεκριμένο μετασχηματισμό τα bytes των γραμμών του State Array ολισθαίνονται κυκλικά. Η πρώτη γραμμή παραμένει όπως είναι, ενώ τα bytes των υπολοίπων ολισθαίνουν με διαφορετικό offset. Όπως φαίνεται και στην **Εικόνα 15**, η δεύτερη γραμμή ολισθαίνει κυκλικά και αριστερόστροφα (left-shifted) κατά ένα byte, η τρίτη γραμμή κατά τον ίδιο τρόπο ολισθαίνει κυκλικά και αριστερόστροφα κατά 2 bytes και τέλος αντίστοιχα η τέταρτη γραμμή ολισθαίνει κυκλικά και αριστερόστροφα κατά 3 bytes.

Καθώς οι μετασχηματισμοί MixColumns και AddRoundKey εφαρμόζονται στήλη προς στήλη, ο μετασχηματισμός ShiftRows εξασφαλίζει ότι τα 4 bytes κάθε στήλης κατανέμονται στις 4 διαφορετικές στήλες. Στην παρακάτω εικόνα αποτυπώνεται ο τρόπος με τον οποίο ο συγκεκριμένος μετασχηματισμός επηρεάζει τον State Array.



Εικόνα 15: *ShiftRows- Cyclically shifts the last 3 rows in the State Array [1]*

Στο σχήμα που φαίνεται παρακάτω, παρουσιάζεται ένα παράδειγμα εφαρμογής του μετασχηματισμού ShiftRows σύμφωνα με όσα αναλύθηκαν παραπάνω.

87	F2	4D	97
EC	6E	4C	90
4A	C3	46	E7
8C	D8	95	A6

→

87	F2	4D	97
6E	4C	90	EC
46	E7	4A	C3
A6	8C	D8	95

4.4 MixColumns Μετασχηματισμός

Ο μετασχηματισμός MixColumns εφαρμόζεται ξεχωριστά σε κάθε στήλη του State Array του αλγορίθμου AES, αντιμετωπίζοντας κάθε στήλη ως ένα πολυώνυμο τρίτου βαθμού. Οι στήλες μπορούν να θεωρηθούν ως πολυώνυμα GF(28) που πολλαπλασιάζονται ως προς x4 + 1 με ένα σταθερό πολυώνυμο a(x), το οποίο δίνεται από τη σχέση

$$a(x) = \{03\}x3 + \{01\}x2 + \{01\}x + \{02\}$$

Κάθε byte μιας στήλης του State Array αντιστοιχίζεται σε μία νέα τιμή που αποτελεί συνδυασμός και των τεσσάρων bytes της στήλης. στην οποία ανήκει, όπως φαίνεται και παρακάτω. Για τη διαδικασία της κρυπτογράφησης ισχύει η σχέση

$$S'(x) = a(x) * S(x)$$

η οποία εκφράζεται σε μορφή μητρώου ως εξής:

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix}$$

Ως αποτέλεσμα της παραπάνω σχέσης, προκύπτουν οι εξής εξισώσεις:

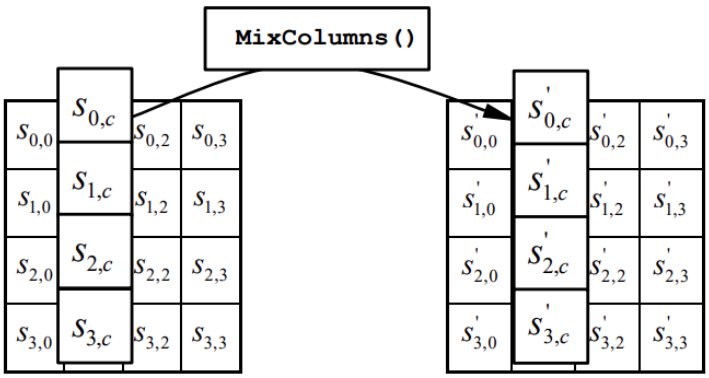
S'0,c = ({02} * S0,c) + ({03} * S1,c) + S2,c + S3,c

S'1,c = ({02} * S1,c) + ({03} * S2,c) + S0,c + S3,c

S'2,c = ({02} * S2,c) + ({03} * S3,c) + S0,c + S1,c

S'3,c = ({02} * S3,c) + ({03} * S0,c) + S1,c +S2,c

Στην **Εικόνα 16** φαίνεται γραφικά η εφαρμογή του μετασχηματισμού MixColumns.



Εικόνα 16: *MixColumns Operation on the State Array Column-by-Column [1]*

Στο παρακάτω σχήμα παρουσιάζεται ένα παράδειγμα του συγκεκριμένου μετασχηματισμού σύμφωνα με όσα αναλύθηκαν παραπάνω.

87	F2	4D	97
6E	4C	90	EC
46	E7	4A	C3
A6	8C	D8	95

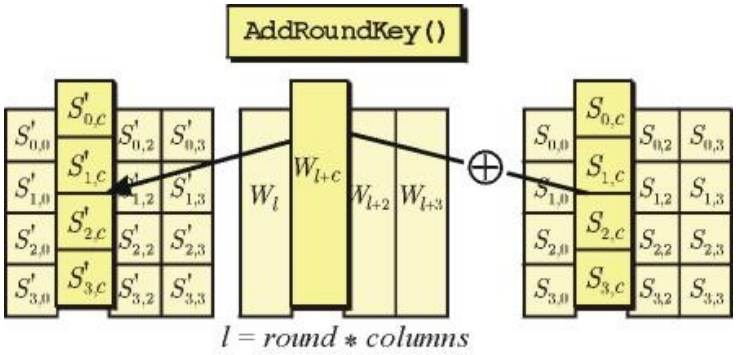
→

47	40	A3	4C
37	D4	70	9F
94	E4	3A	42
ED	A5	A6	BC

4.5 AddRoundKey Μετασχηματισμός

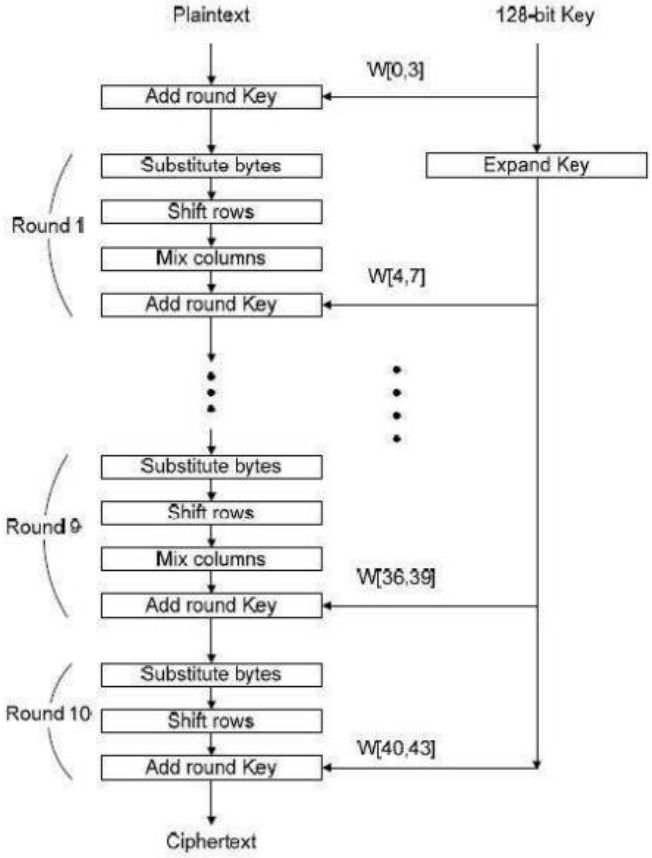
Στον συγκεκριμένο μετασχηματισμό και στα 128 bits του State Array εφαρμόζεται η λογική πράξη XOR σε συνδυασμό με τα τέσσερα 32-bit words του επεκταμένου κλειδιού. Ο μετασχηματισμός AddRoundKey αποτελεί τη μοναδική κρυπτογραφική διαδικασία που συμπεριλαμβάνει τη χρήση του κλειδιού με σκοπό την προστασία των δεδομένων. Η πράξη που αποτελεί τον μετασχηματισμό, θεωρείται ως μια λειτουργία που διενεργείται στήλη προς στήλη μεταξύ των 4 bytes μιας στήλης του State Array και μιας λέξης του round key ή διαφορετικά ως μια λειτουργία σε επίπεδο byte. Η παραπάνω πράξη μεταφράζεται μαθηματικά στην εξής σχέση:

$$[S'_{0,c}, S'_{1,c}, S'_{2,c}, S'_{3,c},] = [S_{0,c}, S_{1,c}, S_{2,c}, S_{3,c},] \{xor\} [W_{round*Nb+c}]$$



Εικόνα 17: *Apply AddRoundKey() to the State Array*

Η λειτουργία forward cipher του AES με μέγεθος κλειδιού 128-bit παρουσιάζεται στην **Εικόνα 18**, όπου το w[i, i+3] αναπαριστά μια ομαδοποίηση ανά 4 λέξεις του επεκταμένου κλειδιού, με $0 \leq i \leq 40$.



Εικόνα 18: *AES Forward Cipher Operation*

Το παρακάτω σχήμα αποτελεί ένα παράδειγμα εφαρμογής του AddRoundKey μετασχηματισμού σύμφωνα με όσα αναλύθηκαν παραπάνω.

47	40	A3	4C
37	D4	70	9F
94	E4	3A	42
ED	A5	A6	BC

⊕

AC	19	28	57
77	FA	D1	5C
66	DC	29	00
F3	21	41	6A

=

EB	59	8B	1B
40	2E	A1	C3
F2	38	13	42
1E	84	E7	D2

Για λόγους πληρότητας αναφέρεται πως η αποκρυπτογράφηση αποτελεί την αντίστροφη διαδικασία της κρυπτογράφησης. Πιο συγκεκριμένα, αντιστρέφει τους μετασχηματισμούς που έχουν αναφερθεί παραπάνω προκειμένου να υπολογίσει το αρχικό plaintext δοθέντος του κρυπτογραφημένου ciphertext. Κατά τη διαδικασία της αποκρυπτογράφησης χρησιμοποιούνται οι εξής μετατροπές: AddRoundKey, InvMixColumns, InvShiftRows και InvSubBytes. Επισημαίνονται παρακάτω τα εξής σημεία:

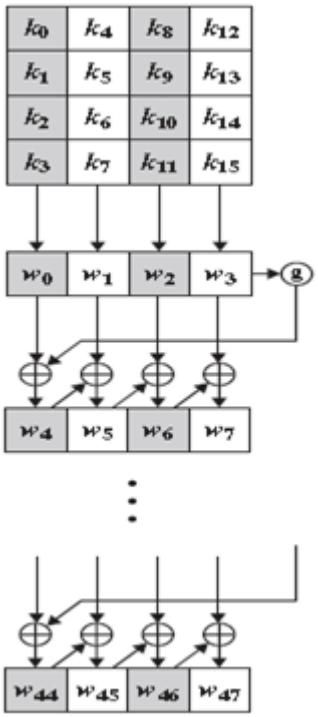
- Ο μετασχηματισμός **AddRoundKey** αποτελεί από μόνος του αντίστροφη συνάρτηση καθώς η πράξη XOR είναι αντιστρέψιμη.
- Ο μετασχηματισμός **InvMixColumns** χρησιμοποιεί διαφορετικό πολυώνυμο από αυτό του χρησιμοποιείται στον MixColumns.
- Ο μετασχηματισμός **InvShiftRows** πραγματοποιεί ολίσθηση των bytes προς τα δεξιά αντί για τα αριστερά.
- Ο μετασχηματισμός **InvSubBytes** αντιστρέφει το look-up table S-box με έναν αντίστροφο affine μετασχηματισμό, ακολουθούμενο από την ίδια αντιστροφή στο GF(2^8) σαν και αυτή που χρησιμοποιήθηκε στην κρυπτογράφηση.

Για περισσότερες πληροφορίες σχετικά με την αποκρυπτογράφηση, προτείνεται το επίσημο paper του NIST [1].

4.5.1 AES Key Expansion

Κατά τη συγκεκριμένη διαδικασία, λαμβάνεται ένα 128-bit κλειδί ως είσοδος και σε κάθε συνεδρία προκύπτει ένα επεκταμένο κλειδί, το οποίο αποτελείται από 44 λέξεις των 32 bits. Κατά τη διάρκεια κάθε γύρου, ο AES cipher χρησιμοποιεί τις 4 από τις 44 λέξεις του επεκταμένου κλειδιού στον μετασχηματισμό AddRoundKey, όπως φαίνεται και στην **Εικόνα 19**. Πιο συγκεκριμένα, οι πρώτες 4 λέξεις του πίνακα εξόδου, που συμβολίζεται ως w, είναι ουσιαστικά τα 16 bytes (128 bits) της αρχικής εισόδου του κρυφού κλειδιού. Πρακτικά, αυτό σημαίνει ότι το αρχικό δοθέν κλειδί αντιγράφεται στις

πρώτες 4 λέξεις του επεκταμένου κλειδιού. Το υπόλοιπο επεκταμένο κλειδί συμπληρώνεται ανά 4 λέξεις κάθε φορά.



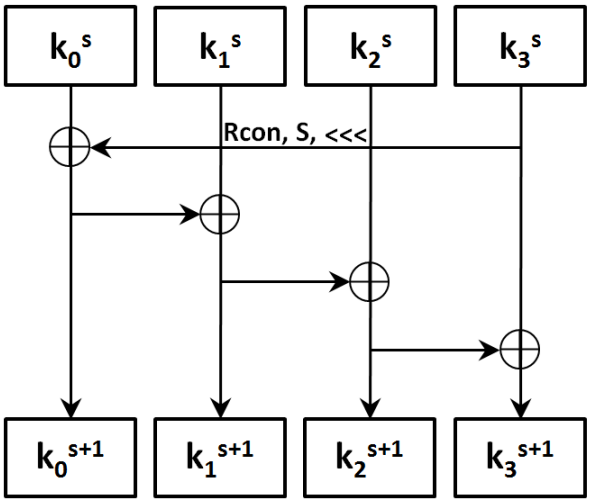
Εικόνα 19: *AES Key Expansion [3]*

Στην **Εικόνα 19** παρουσιάζεται ο τρόπος με τον οποίο πραγματοποιείται η δημιουργία των πρώτων οκτώ λέξεων του επεκταμένου κλειδιού με τη χρήση του συμβόλου g που εκπροσωπεί μία πολύπλοκη λειτουργία. Η συγκεκριμένη λειτουργία αποτελείται από τις ακόλουθες υπολειτουργίες [3]:

- **RotWord**- πραγματοποιεί μια περιστροφή προς τα αριστερά κατά ένα byte σε μια λέξη, με αποτέλεσμα η είσοδος $[B_0, B_1, B_2, B_3]$ σε $[B_1, B_2, B_3, B_0]$
- **SubWord**- πραγματοποιεί την αντικατάσταση (substitution) ενός byte σε κάθε byte του μετατοπισμένου αποτελέσματος, χρησιμοποιώντας S-box. Η συγκεκριμένη λειτουργία είναι αρκετά παρόμοια με αυτή που αναλύθηκε στο μετασχηματισμό SubBytes του κρυπτογραφικού αλγορίθμου AES.
- Στο τελευταίο βήμα, πραγματοποιείται η λογική πράξη XOR μεταξύ του αποτελέσματος των 2 παραπάνω βημάτων και της σταθεράς γύρου (round constant) Rcon[i].

Στην **Εικόνα 20** παρουσιάζεται η λειτουργία ενός single round key computation όπως ακριβώς αναλύθηκε παραπάνω. Η διαδικασία αυτή επαναλαμβάνεται κάνοντας χρήση του round key ως cipher, ώστε να παράγει τα επόμενα 128 bits του επεκταμένου κλειδιού.

Επισημαίνεται πως η τιμή του Rcon[i] αρχικοποιείται με την τιμή 1 και σε κάθε γύρο προσαυξάνεται.



Εικόνα 20: *The Key Schedule of AES*

Η σταθερά γύρου είναι μια λέξη όπου τα τρία δεξιότερα bytes της είναι ίσα με το 0. Αυτό σημαίνει ότι η λογική πράξη XOR μεταξύ μιας λέξης και του Rcon εφαρμόζεται ουσιαστικά μόνο στο αριστερότερο byte της λέξης.

Η σταθερά γύρου είναι διαφορετική για κάθε γύρο και ορίζεται ως:

$$Rcon[i] = (RC[i], 0, 0, 0) \text{ με } RC[1] = 1, RC[i] = 2 * RC[i-1]$$

με τον πολλαπλασιασμό να ορίζεται στο πεδίο $GF(2^8)$. Οι τιμές του RC[i] σε δεκαεξαδικό σύστημα είναι:

j	1	2	3	4	5	6	7	8	9	10
RC[j]	01	02	04	08	10	20	40	80	1B	36

Πίνακας 2: *Round Constant (RC) in Hexadecimal Format [3]*

Για παράδειγμα, αν υποθέσουμε ότι το round key για τον όγδοο γύρο είναι:

EA D2 73 21 B5 8D BA D2 31 2B F5 60 7F 8D 29 2F

Τότε, τα πρώτα 4 bytes του round key για τον ένατο γύρο είναι τα εξής:

i (decimal)	temp	After RotWord	After SubWord	Rcon (9)	After XOR with Rcon	w[i 4]	w[i] = temp ⊕ w[i 4]
36	7F8D292F	8D292F7F	5DA515D2	1B000000	46A515D2	EAD27321	AC7766F3

Οι Rijndael σχεδίασαν τον AES Key Expansion με σκοπό να αντιμετωπίσουν γνωστές κρυπταναλυτικές επιθέσεις. Το παραπάνω επιτυγχάνεται με τη προσθήκη μιας σταθεράς γύρου, η οποία εξαρτάται αποκλειστικά από τον γύρο και έχει ως αποτέλεσμα την εξάλειψη της συμμετρίας ή της ομοιότητας μεταξύ των τρόπων παραγωγής κλειδιών γύρου σε διαφορετικούς γύρους. Τα κριτήρια που χρησιμοποιήθηκαν, περιγράφονται παρακάτω [3]:

- Η γνώση ενός τμήματος του κλειδιού ή του γύρου δεν επιτρέπει τον υπολογισμό πολλών άλλων bits του γύρου.
- Αποτελεί αντιστρέψιμη διαδικασία, δηλαδή η γνώση οποιωνδήποτε N_k συνεχόμενων λέξεων του επεκταμένου κλειδιού επιτρέπει την επαναδημιουργία ολόκληρου του επεκταμένου κλειδιού (N_κ = μέγεθος κλειδιού σε λέξεις).
- Η Ταχύτητα σε ευρύ φάσμα επεξεργαστών.
- Η Χρήση σταθερών γύρων για την εξάλειψη συμμετριών.
- Η Διάχυση των cipher key διαφορών μέσα στα round keys, το οποίο υποδεικνύει ότι κάθε key bit επηρεάζει πολλά round key bits.
- Η Επαρκής μη γραμμικότητα, προκειμένου να αποφευχθεί ο πλήρης καθορισμός των round key διαφορών μόνο από τις διαφορές του cipher key
- Η απλότητα της εφαρμογής

Οι συγγραφείς δεν παρέχουν παραπάνω ποσοτικές πληροφορίες σχετικά με το πόσα ακριβώς bits απαιτούνται όσον αφορά στο πρώτο bullet της προηγούμενης λίστας. Ωστόσο, η ιδέα που εκφράζεται, είναι ότι, αν κανείς έχει λιγότερα από N_k διαδοχικές λέξεις του κλειδιού κρυπτογράφησης ή του κλειδιού γύρου, τότε είναι αρκετά δύσκολο να ανακατασκευάσει τα υπόλοιπα άγνωστα bits. Αυτό σημαίνει ουσιαστικά ότι όσα λιγότερα bits είναι γνωστά, τόσο πιο δύσκολο είναι να προχωρήσει κανείς στην ανακατασκευή ή τον προσδιορισμό άλλων bits του επεκταμένου κλειδιού.

5

Galois/ Counter Mode (GCM)

Ο Galois/Counter Mode (GCM) σχεδιάστηκε για να ανταποκρίνεται στην ανάγκη για έναν πιστοποιημένο τρόπο κρυπτογράφησης που μπορεί αποτελεσματικά να υποστηρίξει ταχύτητες 10 gigabits ανά δευτερόλεπτο και ακόμα υψηλότερες, όταν χρησιμοποιείται σε υλικό. Επιπλέον, σχεδιάστηκε να παρουσιάζει αξιόλογες επιδόσεις και στον τομέα του λογισμικού, ενώ παράλληλα παραμένει απαλλαγμένο από πνευματικούς περιορισμούς ιδιοκτησίας. Η απόδοση του GCM το καθιστά ιδιαίτερα χρήσιμο για εφαρμογές που απαιτούν υψηλές ταχύτητες και ασφάλεια.

Ο λειτουργικός τρόπος Galois/Counter Mode (GCM) ενσωματώνει τον CTR [9] και τον υποστηρίζει προσθέτοντας έναν κώδικα αυθεντικοποίησης μηνύματος (MAC) βασισμένο σε μία καθολική hashing. Η βασική λειτουργία περιλαμβάνει πολλαπλασιασμό στο πεπερασμένο πεδίο $GF(2^w)$, όπου η βασική λειτουργία είναι ο πολλαπλασιασμός με ένα σταθερό στοιχείο πεδίου (fixed field element). Ο GCM μπορεί να υλοποιηθεί αποτελεσματικά σε λογισμικό μέσω μεθόδων οδηγούμενων από πίνακα (table-driven methods). Είναι σημαντικό να σημειωθεί ότι ο GCM μπορεί να χρησιμοποιηθεί ως αυτόνομο MAC ή ως πρόσθετο MAC.

5.1 Τα Στοιχειώδη Μέρη του GCM

5.1.1 Block Cipher

Στο πρότυπο AES-GCM, οι λειτουργίες που περιγράφονται εξαρτώνται από την επιλογή ενός βασικού συμμετρικού κλειδιού block cipher AES και, συνεπώς, μπορούν να θεωρηθούν ως κατάσταση

λειτουργίας (mode of operation) του block cipher. Το AES-GCM κλειδί είναι το ίδιο με το κλειδί του block cipher που χρησιμοποιείται, το οποίο χρησιμοποιείται και για τις διάφορες λειτουργίες που περιλαμβάνει το πρωτόκολλο AES-GCM, όπως είναι οι λειτουργίες κρυπτογράφησης και αυθεντικοποίησης.

Για κάθε δοσμένο κλειδί, το αντίστοιχο block cipher του mode περιλαμβάνει δύο λειτουργίες που είναι αντίστροφες μεταξύ τους. Η επιλογή του block cipher περιλαμβάνει την ονομασία μίας από τις δύο λειτουργίες του block cipher ως η "forward cipher function," όπως και στις προδιαγραφές του αλγορίθμου AES. Στην περίπτωση του AES-GCM, η forward cipher function αναφέρεται στη λειτουργία κρυπτογράφησης (encryption function) του AES block cipher, η οποία είναι η λειτουργία που πραγματοποιείται στο ECB mode [9]. Αξίζει να σημειωθεί ότι ο GCM δεν χρησιμοποιεί την inverse cipher function του AES για τις λειτουργίες του.

Η "forward cipher function" είναι μια αναδιάταξη (permutation) πάνω σε bit strings σταθερού μήκους, με τα strings να ονομάζονται "μπλοκ." Το μήκος ενός μπλοκ ονομάζεται "μέγεθος μπλοκ" (block size). Το κλειδί παριστάνεται ως K, και η προκύπτουσα forward cipher λειτουργία του κρυπτογραφικού αλγορίθμου χαρακτηρίζεται ως $CIPH_K$.

Για το βασικό block cipher στο πλαίσιο του GCM, ισχύουν τα εξής:

1. Το βασικό block cipher πρέπει να είναι εγκεκριμένο.
2. Το μέγεθος του μπλοκ πρέπει να είναι 128 bits.
3. Το μέγεθος του κλειδιού πρέπει να είναι τουλάχιστον 128 bits.

Το κλειδί πρέπει να παράγεται τυχαία με ομοιόμορφο τρόπο ή σχεδόν ομοιόμορφο τρόπο, εξασφαλίζοντας ότι κάθε δυνατό κλειδί είναι (σχεδόν) εξίσου πιθανό να δημιουργηθεί. Ως εκ τούτου, το κλειδί θα πρέπει να είναι "φρέσκο" (fresh), δηλαδή, άνισο με κάθε προηγούμενο κλειδί, με μεγάλη πιθανότητα. Επιπλέον, το κλειδί πρέπει να εγκαθίσταται μυστικά μεταξύ των εμπλεκόμενων φορέων που συμμετέχουν στην επικοινωνία και να χρησιμοποιείται αποκλειστικά για το GCM με το επιλεγμένο block cipher. Πρόσθετες απαιτήσεις σχετικά με τη δημιουργία και τη διαχείριση των κλειδιών συζητούνται σε παρακάτω ενότητα..

5.1.2 Οι Δύο GCM Λειτουργίες

Το GCM αποτελείται από δύο λειτουργίες: την πιστοποιημένη κρυπτογράφηση και πιστοποιημένη αποκρυπτογράφηση. Πιο συγκεκριμένα,

1. **Πιστοποιημένη Κρυπτογράφηση (Authenticated Encryption)**:
    - Κρυπτογραφεί τα confidential data.

- Υπολογίζει ένα authentication tag (T) που είναι κοινό για τα confidential data και τα Additional Authenticated Data (AAD).

2. **Πιστοποιημένη Αποκρυπτογράφηση (Authenticated Decryption)**:
    - Αποκρυπτογραφεί τα confidential data.
    - Επαληθεύει τη γνησιότητα τους μέσω του authentication tag.

Σε περιπτώσεις που έχουμε μόνο Additional Authenticated Data (AAD) χωρίς confidential data, η εκδοχή του GCM ονομάζεται GMAC. Συνεπώς, η λειτουργία GMAC προσφέρει υπολογισμό και επιβεβαίωση του Tag μόνο για τα non-confidential data.

5.1.2.1 Πιστοποιημένη Κρυπτογράφηση

1. **Input Data**

Η λειτουργία της πιστοποιημένης κρυπτογράφησης (Authenticated Encryption) στο πλαίσιο του GCM χρησιμοποιεί τρία strings εισόδου:

- **Plaintext (Κείμενο Καθαρού Κειμένου)-** αυτό είναι το κείμενο που πρέπει να κρυπτογραφηθεί, συμβολίζεται ως P, και μπορεί να έχει μήκος από 0 έως 239- 256 bits.
- **Additional Authentication Data (AAD- Επιπρόσθετα Δεδομένα Πιστοποίησης)**- αυτά τα δεδομένα, συμβολίζονται ως A, μπορούν να έχουν μήκος από 0 έως 264 bits και προστίθενται στον υπολογισμό του authentication tag.
- **Initialization Vector (IV- Διάνυσμα Αρχικοποίησης)**- Το IV είναι ουσιαστικά μια τυχαία τιμή, γνωστή και ως nonce, με μήκος μεταξύ 1 και 264- 1 bits. Καθορίζει μοναδικά κάθε κλήση της λειτουργίας κρυπτογράφησης για τα δεδομένα που πρέπει να προστατευτούν.

Είναι σημαντικό να εξασφαλίζεται η μοναδικότητα του IV (και του κλειδιού), όπως περιγράφεται αναλυτικά σε παρακάτω ενότητα.

Το GCM προστατεύει τόσο την αυθεντικότητα (authenticity) όσο και την εμπιστευτικότητα (confidentiality) των δεδομένων. Επιβεβαιώνει τη γνησιότητα του απλού κειμένου (plaintext- P) και των επιπρόσθετων δεδομένων πιστοποίησης (AAD), προσφέροντας παράλληλα εμπιστευτικότητα του P, ενώ τα AAD μεταδίδονται χωρίς την ανάγκη κρυπτογράφησης.

Σημειώνεται πως, παρόλο που ο GCM ορίζεται για bit strings, τα μήκη του plaintext, των AAD, και του διανύσματος αρχικοποίησης (IV) πρέπει να είναι πολλαπλάσια του 8, ώστε να αντιστοιχούν σε byte strings.

2. Output Data

Τα ακόλουθα δύο bit strings που παρέχονται, αποτελούν τα δεδομένα εξόδου σε μια λειτουργία πιστοποιημένης κρυπτογράφησης.

- **Κρυπτοκείμενο (Ciphertext)**- συμβολίζεται ως c και έχει ακριβώς το ίδιο μήκος με το αρχικό κείμενο (plaintext).
- **Authentication Tag**- συμβολίζεται ως T, με μήκος που μπορεί να είναι μία από τις τιμές 128, 120, 112, 104, 96, 64 ή 32.

Το μήκος του tag, συμβολίζεται ως t, και καθορίζει τη δύναμη της αυθεντικότητας των δεδομένων εισόδου P, του Authentication tag A, και του Initialization Vector (IV).

Το μήκος του tag αποτελεί παράμετρο ασφάλειας και η χρήση συγκεκριμένων τιμών πρέπει να συμμορφώνεται με τις οδηγίες που παρέχονται από το NIST, όπως αναφέρεται στα παραρτήματα Β και Γ [9]. Επιπλέον, μια εφαρμογή πρέπει να περιορίσει το υποστηριζόμενο μήκος του tag σε μία από τις επτά επιλογές που αναφέρθηκαν, και μια σταθερή τιμή για το t πρέπει να συνδέεται με κάθε κλειδί, λαμβάνοντας υπόψη τις απαιτήσεις σχετικά με το μήκος των δεδομένων και τη διάρκεια ζωής του κλειδιού.

5.1.2.2 Authenticated Decryption

Κατά τη λειτουργία της πιστοποιημένης αποκρυπτογράφησης, με δεδομένα το εγκεκριμένο block cipher. το κλειδί και το αντίστοιχο μήκος του tag, οι είσοδοι περιλαμβάνουν τις τιμές των:

- Initialization Vector (IV)
- Ciphertext (C)
- Additional authentication data (AAD) (A)
- Authentication tag (T)

Όπως αυτά αναφέρθηκαν παραπάνω.

Εάν η αποκρυπτογράφηση είναι επιτυχής και το authentication tag T' που υπολογίστηκε είναι ίδιο με το παρεχόμενο authentication tag T, τότε η έξοδος είναι το Plaintext P που αντιστοιχεί στο Ciphertext C. Σε αντίθετη περίπτωση, η έξοδος είναι ένας ειδικός κωδικός σφάλματος (Fail), που υποδεικνύει ότι τα δεδομένα εισόδου δεν ήταν γνήσια.

Επίσης, το FAIL παράγεται εάν τα δεδομένα εισόδου δεν έχουν κρυπτογραφηθεί με το συγκεκριμένο κλειδί. Οι τιμές των len(C), len(A), και len(IV) πρέπει να είναι ίδιες με τις τιμές των len(P), len(A), και len(IV) που υποστηρίζει η εφαρμογή για τη λειτουργία πιστοποιημένης κρυπτογράφησης.

5.1.3 Εμπιστευτικότητα και Πιστοποίηση

Η προστασία του απορρήτου του plaintext εντός του GCM επιτυγχάνεται μέσω μιας παραλλαγής του Counter mode, χρησιμοποιώντας μια συγκεκριμένη προσαυξητική συνάρτηση για τη δημιουργία των counter blocks. Το πρώτο counter block για την κρυπτογράφηση του plaintext παράγεται από ένα block που αυξάνεται και δημιουργείται από το Initialization Vector (IV).

Ο μηχανισμός πιστοποίησης εντός του GCM βασίζεται σε μια hash συνάρτηση, την GHASH, η οποία περιγράφει τον πολλαπλασιασμό με μια σταθερή παράμετρο, γνωστή ως δευτερεύον κλειδί hash (hash subkey), μέσα σε ένα δυαδικό Galois πεδίο.

Το δευτερεύον κλειδί hash, συμβολίζεται ως H, προκύπτει από το block cipher με είσοδο το "μηδενικό" block. Το αποτέλεσμα, γνωστό ως $GHASH_H$, χρησιμοποιείται για τη συμπίεση μιας κωδικοποίησης των AAD και ciphertext σε ένα ενιαίο block, το οποίο κρυπτογραφείται για τη δημιουργία του authentication tag.

Η GHASH λειτουργεί ως keyed hash λειτουργία και αποτελεί απαίτηση για το GCM, αποκλείοντας άλλες κρυπτογραφικές hash λειτουργίες. Ενδιάμεσες τιμές κατά την εκτέλεση του GCM θεωρούνται απόρρητες. Συγκεκριμένα, αυτή η απαίτηση αποκλείει τον GCM από το να χρησιμοποιείται με το hash subkey δημοσίως για άλλους σκοπούς, εξασφαλίζοντας την απόρρητη φύση των ενδιάμεσων τιμών.

5.1.4 Τύπο Εφαρμογών του GCM

Σύμφωνα με το SP800-380D, υπάρχουν τέσσερις τύποι εφαρμογών του GCM:

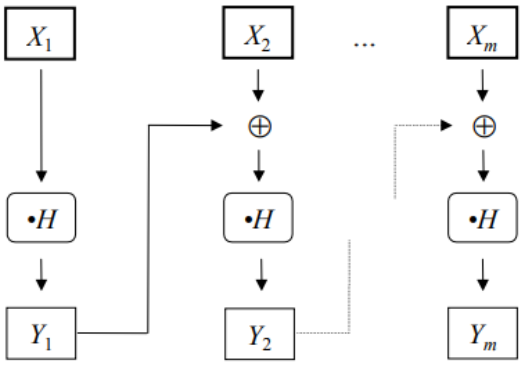
1. **GCM με αυθαίρετου μεγέθους IV**: Σε αυτήν την εφαρμογή, το Initialization Vector (IV) μπορεί να έχει οποιοδήποτε μέγεθος.
2. **GCM με καθορισμένου μεγέθους (default) IV**: Σε αυτήν την εφαρμογή, το μέγεθος του IV περιορίζεται ακριβώς στα 96 bits.
3. **GMAC**: Αυτός ο τύπος παράγει ένα ανεξάρτητο authentication tag T για τα Additional Authentication Data (AAD) με IV αυθαίρετου μεγέθους. Το plaintext P είναι το κενό string.
4. **GMAC με καθορισμένο IV**: Σε αυτήν την εφαρμογή επιλέγεται η χρήση του GCM με καθορισμένο (default) IV, το οποίο περιορίζεται στα 96 bits.

5.2 Τα Βασικά Μαθηματικά του GCM

5.2.1 GHASH Function

Στον GCM, ο μηχανισμός πιστοποίησης (authentication mechanism) βασίζεται σε μια hash function που ονομάζεται GHASH. Αυτή η λειτουργία πραγματοποιεί πολλαπλασιασμό με μια σταθερή παράμετρο που αποκαλείται hash subkey (H) μέσα στο δυαδικό Galois Field GF(2^{128}). Το hash subkey παράγεται εφαρμόζοντας το block cipher σε ένα μπλοκ μηδενικών bits, και το αποτέλεσμα είναι το $GHASH_H$. Η λειτουργία GHASH χρησιμοποιείται για να συμπιέσει μια κωδικοποίηση των Additional Authentication Data (AAD) και του ciphertext σε ένα μόνο μπλοκ. Στη συνέχεια, αυτό το μπλοκ κρυπτογραφείται, παράγοντας έτσι το authentication tag

Είναι σημαντικό να σημειωθεί ότι η GHASH λειτουργεί ως μια hash function κλειδιού, αλλά δεν αποτελεί από μόνη της μια κρυπτογραφική hash function. Ο βασικός ρόλος της GHASH είναι να παρέχει μοναδικές υπογραφές για το AAD και το ciphertext, ενισχύοντας έτσι την αυθεντικότητα των δεδομένων.



Εικόνα 21: *$GHASH_H$ (X1 || X2 || ... || Xm) = Ym [8]*

Ουσιαστικά, η GHASH Function υπολογίζει το:

$$X_1 \bullet H^m \{xor\} X_2 \bullet H^{m-1} \{xor\} \dots \{xor\} X_{m-1} H^2 \{xor\} X_m \bullet H$$

Η GHASH function φαίνεται στην **Εικόνα 21** χωρίς το zero block Y_0 αφού αυτό όταν γίνει exclusive-OR με το X_1, το τελευταίο δεν αλλάζει.

Για καλύτερη κατανόηση, παρακάτω δίνεται η λειτουργία της GHASH Function υπό τη μορφή ψευδοκώδικα.

Algorithm 1

$Y \leftarrow 0$

for i = 1 to m do

$Y \leftarrow (Y \{xor\} X) \bullet H$

end for

return Y

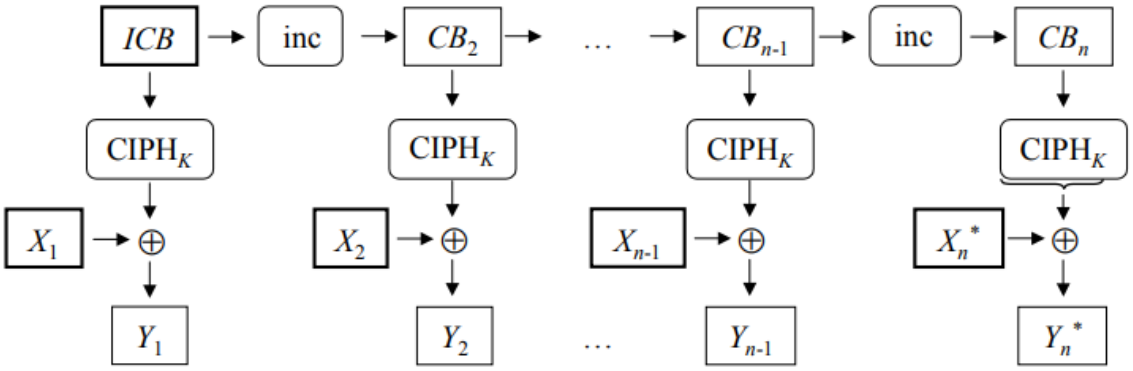
Η GHASH ουσιαστικά πρόκειται για μια σειριακή λειτουργία, καθώς κάθε βαθμίδα υπολογίζεται με βάση το αποτέλεσμα της προηγούμενης βαθμίδας. Η διαδοχική εξάρτηση μεταξύ των βαθμίδων αυτών αποκλείει την δυνατότητα παραλληλισμού των υπολογισμών, καθώς κάθε βαθμίδα πρέπει να περιμένει το αποτέλεσμα της προηγούμενης. Επιπλέον, η καθυστέρηση της GHASH εξαρτάται από την καθυστέρηση του πολλαπλασιαστή, και επομένως η απόδοση της είναι περίπλοκη και επηρεάζεται από την επιλογή του υλικού. Συνεπώς, η επιλογή ενός γρήγορου και αποδοτικού πολλαπλασιαστή είναι κρίσιμη για την ταχύτητα και την απόδοση του AES-GCM. Συμπερασματικά, η παραλληλοποίηση και η επιλογή γρήγορων υλικών είναι ουσιώδης για τη βελτίωση της απόδοσης του AES-GCM, ιδίως όταν αντιμετωπίζουμε ζητήματα ταχύτητας σε εφαρμογές κρυπτογράφησης.

5.2.2 GCTR Function

Ο μηχανισμός εμπιστευτικότητας (confidentiality) του GCM βασίζεται σε μια παραλλαγή του Counter mode, που ονομάζεται GCTR (Galois Counter Mode). Στον GCTR, χρησιμοποιείται μια ειδική συνάρτηση αυξήσεως (incrementing function), συμβολίζεται ως inc και παράγει την ακολουθία των counter blocks. Η διαδικασία ξεκινά με τον υπολογισμό του πρώτου counter block, το οποίο προκύπτει αυξάνοντας ένα block που προέρχεται από τον Initialization Vector (IV). Κάθε επόμενο counter block υπολογίζεται αυξάνοντας το προηγούμενο counter block με τη χρήση της συνάρτησης αύξησης inc. Αυτή η ακολουθία counter blocks χρησιμοποιείται στη συνέχεια για την κρυπτογράφηση του κειμένου (plaintext) με τη χρήση της πράξης XOR (exclusive OR) ως μέρος της διαδικασίας του GCM. Η επιλογή του προηγούμενου counter block για τον υπολογισμό του επόμενου διασφαλίζει τη μοναδικότητα των counter blocks και εξασφαλίζει την ασφάλεια του συστήματος κρυπτογράφησης.

Στο Step 1, το string εισόδου, το οποίο είναι αυθαίρετου μεγέθους, διαιρείται σε μια ακολουθία από blocks των 128 bits. Κατά αυτό τον τρόπο εξασφαλίζεται ότι μόνο το δεξιότερο block της σειράς μπορεί να είναι είτε ένα πλήρες block είτε ένα μη κενό μερικό block. Στα Steps 2, 3, 4, η συνάρτηση αύξησης inc_{32} εφαρμόζεται επαναληπτικά στον Initial Counter Block (ICB), παράγοντας μια σειρά από Counter Blocks (CB). Στα Steps 5 και 6, ο block cipher εφαρμόζεται σε κάθε Counter Block, και το αποτέλεσμα

που προκύπτει, γίνεται XORed με τα αντίστοιχα blocks του string εισόδου. Στο Step 7, η ακολουθία των αποτελεσμάτων που προέκυψαν συνενώνεται για να παραχθεί η έξοδος. Αυτή η διαδικασία επαναλαμβάνεται για κάθε block του string εισόδου, και η τελική έξοδος παράγεται από τη συνένωση των αποτελεσμάτων.



Εικόνα 22: *$GCTR_K$ (ICB, X_1 || X_2 || ... || X_n^*) = Y_1 || Y_2 || ... || Y_n^* [8]*

5.3 GCM Specification

Οι Αλγόριθμοι για την πιστοποιημένη κρυπτογράφηση και αποκρυπτογράφηση του GCM περιγράφονται στις παρακάτω ενότητες. Αυτές οι προδιαγραφές περιλαμβάνουν τις εισόδους, εξόδους, τα βήματα, τα διαγράμματα και μια περιγραφή των αλγορίθμων. Οι προτεινόμενοι αλγόριθμοι δεν καθορίζουν συγκεκριμένο block cipher, αλλά στη συγκεκριμένη διπλωματική χρησιμοποιείται ο AES-GCM. Οι είσοδοι, συχνά αναφερόμενες ως προϋποθέσεις, πρέπει να πληρούν τις απαιτήσεις των ενοτήτων 5.1.2.1 και 5.1.2.2.

5.3.1 Authenticated Encryption

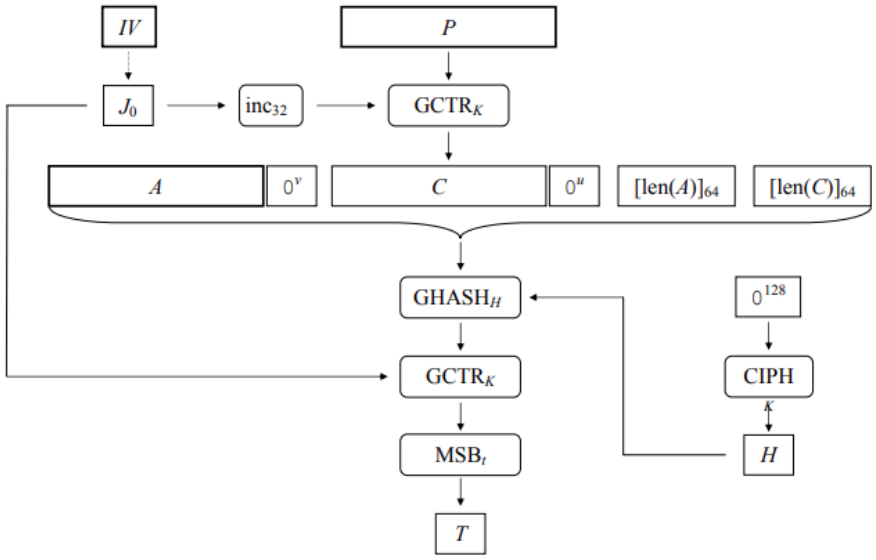
Στο βήμα 1, το hash subkey H για την GHASH Function παράγεται εφαρμόζοντας το block cipher στο zero block. Στο βήμα 2, το pre-counter block (J) παράγεται από τον IV. Αφού το μήκος του IV περιορίζεται αυστηρά στα 96 bits, προστίθεται το padding string $0^{31}||1$ στον IV για να σχηματιστεί το pre-counter block των 128 bits. Στο βήμα 3, η 32-bit incrementing function εφαρμόζεται στο pre-counter block για να δημιουργηθεί το initial counter block (ICB) για μια κλήση της GCTR function για το plaintext. Το αποτέλεσμα της κλήσης της GCTR function είναι το ciphertext.

Στα βήματα 4 και 5, σε καθένα από τα AAD και ciphertext προστίθενται ο ελάχιστος αριθμός μηδενικών bits, έτσι ώστε το bit length των τερματικών strings να είναι πολλαπλάσιο του block size. Η συνένωση

αυτών των strings προστίθεται με την 64-bit αναπαράσταση του μήκους των AAD και ciphertext, και η GHASH function εφαρμόζεται στο αποτέλεσμα για να παράγει ένα single block εξόδου.

Στο βήμα 6, αυτό το block εξόδου κρυπτογραφείται χρησιμοποιώντας την GCTR function και το pre-counter block που παρήχθη στο βήμα 2, και το αποτέλεσμα περικόπτεται στο καθορισμένο μήκος του tag για να σχηματίσει το authentication tag.

Η authenticated encryption function φαίνεται στην **Εικόνα 23** παρακάτω.



Εικόνα 23: *GCM-AE$_K$ (IV, P, A) = (C, T) [8]*

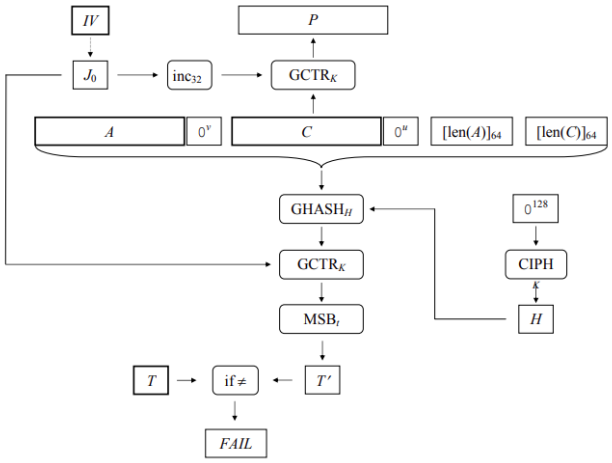
5.3.2 Authenticated Decryption

Στο βήμα 1, το hash subkey H για την GHASH function παράγεται εφαρμόζοντας το block cipher στο zero block. Στο βήμα 2, το pre-counter block (J) παράγεται όπως αναφέρθηκε και στην authenticated encryption function (βήμα 2). Στο βήμα 3, η 32-bit incrementing function εφαρμόζεται στο pre-counter block για να δημιουργηθεί το initial counter block (ICB) για μια κλήση της GCTR function όσον αφορά στο ciphertext αυτή τη φορά. Το αποτέλεσμα της κλήσης της GCTR function είναι το plaintext που αντιστοιχεί στο ciphertext για τον δοσμένο IV.

Στα βήματα 4 και 5, καθένα από τα AAD και το ciphertext προστίθενται με τον ελάχιστο αριθμό μηδενικών bits, πιθανώς και κανένα, έτσι ώστε το bit length των τερματικών strings να είναι πολλαπλάσιο του block size. Η συνένωση αυτών των strings προστίθεται με την 64-bit αναπαράσταση του μήκους των AAD και ciphertext, και η GHASH function εφαρμόζεται στο αποτέλεσμα για να παράγει ένα single block εξόδου.

Στο βήμα 6, αυτό το block εξόδου κρυπτογραφείται χρησιμοποιώντας την GCTR function με το pre-counter block που παρήχθη στο βήμα 2, και το αποτέλεσμα περικόπτεται στο καθορισμένο μήκος του tag για να σχηματίσει το authentication tag. Στο βήμα 7, το αποτέλεσμα του βήματος 6 συγκρίνεται με το authentication tag που έχει ληφθεί από την είσοδο, και αν είναι πανομοιότυπα τότε επιστρέφεται το plaintext, αλλιώς επιστρέφεται το indication FAIL.

Η authenticated decryption function παρουσιάζεται στην **Εικόνα 24** παρακάτω.



Εικόνα 24: *AES-GCM-AD$_K$ (IV, C, A, T) = P or FAIL*

5.4 Απαιτήσεις Μοναδικότητας IV και Keys

Η απαίτηση για μοναδικότητα των Initialization Vectors (IVs) στον GCM είναι κρίσιμη για την ασφάλεια του αλγορίθμου. Πιο συγκεκριμένα, η πιθανότητα να κληθεί η authenticated encryption function με το ίδιο IV και το ίδιο κλειδί για δύο ή περισσότερα διαφορετικά σύνολα δεδομένων εισόδου δεν πρέπει να είναι μεγαλύτερη από 2^{-32}. Η συμμόρφωση με αυτή την απαίτηση είναι σημαντική ώστε να αποτραπούν επιθέσεις πλαστογραφίας [3] που μπορεί να οδηγήσουν σε προβλήματα ασφάλειας. Αν ένα IV επαναληφθεί, μπορεί να υπάρξει αποκάλυψη πληροφοριών και αδυναμία της προστασίας που προσφέρει ο αλγόριθμος.. Είναι σημαντικό επομένως να διασφαλίζεται ότι τα IV είναι μοναδικά για κάθε κλήση του αλγορίθμου με ένα συγκεκριμένο κλειδί.

5.4.1 Επιλογή Κλειδιού

Η απαίτηση για τη μοναδικότητα των GCM κλειδιών επιδιώκει να διασφαλίσει ότι κάθε κλειδί, εγκατεστημένο μεταξύ των προβλεπόμενων χρηστών, πρέπει να είναι πιθανότατα φρέσκο. Για την επίτευξη του παραπάνω βέβαια χρειάζεται ο μηχανισμός αναπαραγωγής κλειδιού (key generation

mechanism) να είναι ανθεκτικός σε παραβιάσεις. Η ανθεκτικότητα αυτή απαιτεί αυστηρή διαχείριση του μηχανισμού παραγωγής του κλειδιού. Εάν ο βασικός μηχανισμός αναπαραγωγής είναι ντετερμινιστικός, τότε πρέπει να παρέχει ισχυρή διαβεβαίωση ότι δεν υπάρχει δυνατότητα επανάληψης των set εισόδων ή των εξόδων από εξωγενείς οντότητες. Για παράδειγμα, τα GCM κλειδιά μπορεί να παράγονται με βάση τη χρήση βασικών λειτουργιών από πρωτόκολλα όπως το Transport Layer Security, Internet Key Exchange v1 και v2, και Secure Shell [10]. Αυτό διασφαλίζει την ανθεκτικότητα του μηχανισμού παραγωγής του κλειδιού έναντι πιθανών επιθέσεων και παρέχει εγγυήσεις για τη μοναδικότητα των κλειδιών.

Κατά τον ίδιο τρόπο, για τη μεταφορά ή διανομή ενός νέου κλειδιού στους αποδέκτες, είναι κρίσιμο να υιοθετηθεί ένας τρόπος που εξασφαλίζει ισχυρή διαβεβαίωση κατά της επανάληψης. Αυτό σημαίνει ότι κανένα μέρος δεν πρέπει να έχει τη δυνατότητα να προκαλέσει την αντικατάσταση ενός προηγούμενου κλειδιού με το κλειδί που προορίζεται για συγκεκριμένη εφαρμογή. Για τα GCM κλειδιά, η εγκατάστασή τους πρέπει να γίνεται στο πλαίσιο ενός εγκεκριμένου κλειδιού διοικητικής δομής. Αυτό εξασφαλίζει όχι μόνο τη φρεσκάδα τους αλλά και την εμπιστευτικότητα και αυθεντικότητά τους.

5.4.2 Κατασκευές IV

Η παρούσα σύσταση παρέχει δύο πλαίσια για την κατασκευή IVs. Το πρώτο πλαίσιο, που περιγράφεται στην ενότητα 5.4.2.1, βασίζεται σε ντετερμινιστικά στοιχεία για την εξασφάλιση της μοναδικότητας. Το δεύτερο πλαίσιο, που περιγράφεται στην ενότητα 5.4.2.2, βασίζεται σε μια μεγάλη συμβολοσειρά εξόδου από εγκεκριμένο Random Bit Generator (RBG) με αρκετή ασφάλεια.

Για μήκος IV έως 96 bits, συνιστάται η χρήση του πρώτου πλαισίου (ντετερμινιστική κατασκευή) σε όλες τις περιπτώσεις χρήσης πιστοποιημένης κρυπτογράφησης με το συγκεκριμένο κλειδί.

Για IV με μήκος 96 bit και άνω, ανάλογα με την υποστήριξη της εφαρμογής, επιλέγεται μόνο ένα από τα δύο πλαίσια: είτε η ντετερμινιστική κατασκευή για τα 96-bit IVs, είτε η RBG-based κατασκευή για τα 128-bit και 160-bit IVs.

Για παράδειγμα, εάν μια εφαρμογή υποστηρίζει IV με μήκη 64 bits, 96 bits, 128 bits και 160 bits, τότε για τα 64-bit IVs η μόνη επιλογή είναι η ντετερμινιστική κατασκευή. Για τα υπόλοιπα μήκη IV, προτείνεται ο συνδυασμός ντετερμινιστικής κατασκευής για 96-bit IVs και RBG-based κατασκευής για 128-bit και 160-bit IVs.

5.4.2.1 Ντετερμινιστική Κατασκευή

Στη ντετερμινιστική κατασκευή, το Initialization Vector (IV) αποτελείται από δύο πεδία: το σταθερό πεδίο (fixed field) και το πεδίο επίκλησης (invocation field). Το σταθερό πεδίο καθορίζει τη συσκευή ή

γενικότερα το πλαίσιο για την εφαρμογή της πιστοποιημένης κρυπτογράφησης. Αντίστοιχα, το πεδίο επίκλησης καθορίζει το σύνολο των εισόδων για τη λειτουργία της πιστοποιημένης κρυπτογράφησης στη συγκεκριμένη συσκευή.

Για κάθε κλειδί, δύο διαφορετικές συσκευές πρέπει να έχουν διαφορετικό σταθερό πεδίο, και δύο διαφορετικά σύνολα εισόδων για οποιαδήποτε μοναδική συσκευή δεν πρέπει να έχουν το ίδιο πεδίο επίκλησης. Η συμμόρφωση με αυτές τις δύο απαιτήσεις εξασφαλίζει τη μοναδικότητα των IVs.

Αν είναι επιθυμητό, το fixed field μπορεί να αποτελείται από δύο ή περισσότερα μικρότερα πεδία. Το ένα από αυτά τα μικρά πεδία μπορεί να αποτελείται από αυθαίρετα bits, χωρίς να είναι αναγκαστικά ντετερμινιστικό ή μοναδικό για τη συσκευή. Αυτό είναι εφικτό υπό τον όρο ότι τα υπόλοιπα bits διασφαλίζουν τη μη επανάληψη του σταθερού πεδίου σε οποιαδήποτε άλλη συσκευή με το ίδιο κλειδί.

Επίσης, το fixed field μπορεί να συνίσταται από αυθαίρετα bits, ιδίως όταν υπάρχει μόνο ένα πλαίσιο για αναγνώριση, όπως στην περίπτωση που ένα φρέσκο κλειδί περιορίζεται σε μια μόνο συνεδρία ενός πρωτοκόλλου επικοινωνίας. Σε αυτή την περίπτωση, αν οι συμμετέχοντες μοιράζονται ένα κοινό σταθερό πεδίο, το πρωτόκολλο εξασφαλίζει ότι τα πεδία επίκλησης είναι διαφορετικά για διαφορετικά δεδομένα εισόδου.

Για την προώθηση της διαλειτουργικότητας, προτείνεται (αλλά δεν είναι υποχρεωτικό) ότι, για το default μήκος Initialization Vector (IV) 96 bits, τα πρώτα 32 bits (αριστερότερα) του IV να αντιστοιχούν στο σταθερό πεδίο, ενώ τα τελευταία 64 bits (δεξιότερα) να αντιστοιχούν στο πεδίο επίκλησης. Ωστόσο, αυτή η πρόταση δεν είναι υποχρεωτική, και εξαρτάται από τις απαιτήσεις της συγκεκριμένης εφαρμογής ή περιβάλλοντος. Τα μήκη και οι θέσεις του σταθερού πεδίου και του πεδίου επίκλησης πρέπει να καθορίζονται για κάθε υποστηριζόμενο μήκος IV σύμφωνα με τις ανάγκες του συστήματος και των πρωτοκόλλων ασφαλείας.

Η προτεινόμενη διάταξη του IV με ένα σταθερό πεδίο στα πρώτα 32 bits και ένα πεδίο επίκλησης στα τελευταία 64 bits έχει σκοπό να προωθήσει τη συνοχή και τη συμβατότητα μεταξύ διαφορετικών υλοποιήσεων.

5.4.2.2 RBG-based Κατασκευή

Στην RBG-based κατασκευή, το Initialization Vector (IV) αποτελείται από δύο πεδία: το τυχαίο πεδίο (random field) και το ελεύθερο πεδίο (free field). Για κάθε υποστηριζόμενο μήκος IV που χρησιμοποιείται στην RBG-based κατασκευή, το μήκος των αυτών των πεδίων πρέπει να καθορίζεται για τη διάρκεια ζωής του κλειδιού. Ειδικότερα, το μήκος του τυχαίου πεδίου πρέπει να είναι τουλάχιστον 96 bits, ενώ το ελεύθερο πεδίο μπορεί να είναι άδειο.

Για κάθε υποστηριζόμενο μήκος IV, ας συμβολίζεται με r(i) το μήκος του τυχαίου πεδίου. Το τυχαίο πεδίο μπορεί να δημιουργηθεί είτε από ένα string εξόδου των r(i) bits από μια εγκεκριμένη Συνάρτηση Παραγωγής Τυχαίων Μεγεθών (RBG) με επαρκή δύναμη ασφάλειας, είτε από το αποτέλεσμα της εφαρμογής της r(i)-bit προσαυξητικής λειτουργίας στο τυχαίο πεδίο του προηγούμενου IV για το συγκεκριμένο κλειδί. Το string εξόδου από την RBG ονομάζεται άμεση τυχαία σειρά (direct random string), και τα τυχαία πεδία που προκύπτουν από την εφαρμογή της r(i)-bit προσαυξητικής λειτουργίας ονομάζονται διάδοχοι (successors).

Στην RBG-based κατασκευή, δεν υπάρχουν απαιτήσεις όσον αφορά στο περιεχόμενο των bits στο ελεύθερο πεδίο του Initialization Vector (IV). Αντίθετα με τη ντετερμινιστική κατασκευή όπου το σταθερό πεδίο εντοπίζει τη συσκευή, στην RBG-based κατασκευή, οι ποσότητες που μπορεί να υπάρχουν στο ελεύθερο πεδίο δεν απαιτείται να είναι διαφορετικές για κάθε συσκευή.

Για κάθε μήκος IV που σχετίζεται με την RBG-κατασκευή, συνίσταται το ελεύθερο πεδίο να είναι κενό, έτσι ώστε το τυχαίο πεδίο να περιλαμβάνει ολόκληρο το IV.

Επιπλέον, είναι σημαντικό να εξασφαλιστεί ότι τα στιγμιότυπα των Συναρτήσεων Παραγωγής Τυχαίων Μεγεθών (RBG) σε διάφορες συσκευές είναι ανεξάρτητα μεταξύ τους. Αυτό διασφαλίζει ότι η κατανομή των άμεσων τυχαίων σειρών σε όλα τα στιγμιότυπα των RBG αναμένεται να είναι ομοιόμορφη. Για παράδειγμα, εάν η αρχικοποίηση των RBG στιγμιότυπων εξαρτάται μόνο από ένα μυστικό "σπόρο", τότε κάθε στιγμιότυπο πρέπει να ξεκινά με ένα ξεχωριστό "σπόρο".

5.4.3 Εμπόδια στον Αριθμό των Επικλήσεων

Η ακόλουθη απαίτηση αφορά όλες τις υλοποιήσεις που χρησιμοποιούν είτε ντετερμινιστική κατασκευή IVs με μήκος διαφορετικό από 96 bits, είτε RBG-based κατασκευή για οποιοδήποτε μήκος IVs. Πιο συγκεκριμένα, ο συνολικός αριθμός των επικλήσεων της λειτουργίας πιστοποιημένης κρυπτογράφησης με δεδομένο κλειδί δεν πρέπει να υπερβαίνει τα 2^{32}, συμπεριλαμβανομένων όλων των μήκων του IV και όλων των εμφανίσεων της λειτουργίας πιστοποιημένης κρυπτογράφησης με το δεδομένο κλειδί.

Για την RBG-based κατασκευή, η απαίτηση ότι $r(i) \geq 96$ όπου r(i) είναι το μήκος του τυχαίου πεδίου για ένα συγκεκριμένο μήκος IV i, είναι επαρκής για τη διασφάλιση της απαίτησης της μοναδικότητας.

Για τη ντετερμινιστική κατασκευή, τα μήκη των πεδίων επίκλησης και σταθερού πεδίου περιορίζουν τον αριθμό των επικλήσεων και των διακριτικών συσκευών που μπορούν να εφαρμόσουν τη λειτουργία πιστοποιημένης κρυπτογράφησης με δεδομένο κλειδί και με δεδομένο μήκος IV.

Συνολικά, αυτές οι περιοριστικές απαιτήσεις θεσπίζονται για να διασφαλιστεί η ασφάλεια και η αποτελεσματικότητα της πιστοποιημένης κρυπτογράφησης.

6

Βασικές Τεχνολογίες της Υλοποίησης

Σε αυτό το κεφάλαιο θα γίνει αναφορά στις τεχνολογίες που χρησιμοποιήθηκαν για την υλοποίηση του θέματος της παρούσας διπλωματικής εργασίας.

6.1 SeedLink

Το πρωτόκολλο SeedLink [11] [12] αναφέρεται σε ένα πρωτόκολλο επικοινωνίας που χρησιμοποιείται ευρέως στον τομέα της σεισμολογίας και της παρακολούθησης σεισμικής δραστηριότητας. Πρόκειται για ένα ανοικτό πρωτόκολλο που αναπτύχθηκε με σκοπό να διευκολύνει τη μετάδοση δεδομένων από σεισμικούς σταθμούς προς τα κέντρα παρακολούθησης και επεξεργασίας. Ουσιαστικά, το SeedLink είναι ένα πρωτόκολλο που επιτρέπει τη μεταφορά σεισμικών δεδομένων σε πραγματικό χρόνο από τους αισθητήρες ενός σεισμικού δικτύου προς τα κέντρα επεξεργασίας, επιτρέποντας έτσι την άμεση ανάλυση της σεισμικής δραστηριότητας. Η διασφάλιση της γρήγορης και αξιόπιστης μετάδοσης δεδομένων είναι κρίσιμη για την άμεση αντίδραση σε περιπτώσεις σεισμών και την αποτελεσματική λειτουργία του σεισμικού παρακολούθησης. Επιπλέον, το SeedLink χρησιμοποιείται και για τη διανομή σεισμικών δεδομένων σε διεθνές επίπεδο, συμβάλλοντας στην παγκόσμια προσπάθεια παρακολούθησης και έρευνας σχετικά με τους σεισμούς και τη σεισμική δραστηριότητα.

Οι βασικές τεχνικές λεπτομέρειες περιλαμβάνουν:

1. **Πρωτόκολλο Επικοινωνίας**: Το SeedLink χρησιμοποιεί το πρωτόκολλο TCP/IP για τη μεταφορά δεδομένων. Αυτό εξασφαλίζει σταθερή και αξιόπιστη σύνδεση μεταξύ των σεισμολογικών σταθμών και των κέντρων παρακολούθησης.
2. **Μοντέλο Δεδομένων**: Το SeedLink χρησιμοποιεί ένα ευέλικτο μοντέλο δεδομένων που επιτρέπει τη μετάδοση διαφορετικών τύπων δεδομένων, όπως επίγειες και υπόγειες κινήσεις, επίπεδα επιτάχυνσης, και άλλες παραμέτρους που σχετίζονται με τη σεισμική δραστηριότητα.
3. **Συμπίεση Δεδομένων**: Το SeedLink συνήθως χρησιμοποιεί τεχνικές συμπίεσης για να μειώσει το εύρος ζώνης και να εξοικονομήσει bandwidth. Η συμπίεση μπορεί να βασίζεται σε διάφορους αλγόριθμους, όπως οι αλγόριθμοι Run-Length Encoding ή GZIP.
4. **Προειδοποίηση και Φιλτράρισμα Δεδομένων**: Το SeedLink μπορεί να προσφέρει δυνατότητες προειδοποίησης για γεγονότα σεισμικής δραστηριότητας και επίσης μπορεί να εφαρμόσει φίλτρα για την επιλεκτική μεταφορά μόνο συγκεκριμένων τύπων δεδομένων.

Το πρωτόκολλο SeedLink μεταδίδει σεισμικά δεδομένα σε μορφή που ονομάζεται MiniSEED (Mini Standard for the Exchange of Earthquake Data). Το MiniSEED είναι ένα διεθνές πρότυπο μορφοποίησης δεδομένων σεισμικής κίνησης, σχεδιασμένο για την αποθήκευση και τη μετάδοση σεισμικών εγγραφών με ελάχιστη χρήση χώρου και εύκολη επεξεργασία.

Κάθε MiniSEED record περιέχει συνήθως τα εξής βασικά χαρακτηριστικά:

1. **Κεφαλίδα (Header)**: Η κεφαλίδα περιέχει πληροφορίες σχετικά με τα δεδομένα που ακολουθούν, όπως το όνομα του σταθμού, οι γεωγραφικές συντεταγμένες, οι παράμετροι εγγραφής, και άλλα μεταδεδομένα.
2. **Δεδομένα (Data Payload)**: Τα πραγματικά δεδομένα κίνησης είναι αποθηκευμένα σε αυτό το τμήμα εγγραφής. Τα δεδομένα μπορεί να είναι επίγειες ή υπόγειες κινήσεις, ανάλογα με τις δυνατότητες του σεισμικού σταθμού.
3. **Σφραγίδα (Trailer)**: Η σφραγίδα περιέχει έναν κωδικό ελέγχου για την εγγραφή, ο οποίος χρησιμοποιείται για την επαλήθευση της ακεραιότητας των δεδομένων.

Είναι σημαντικό να σημειωθεί ότι τα MiniSEED records είναι σχεδιασμένα για αποθήκευση δεδομένων σε πραγματικό χρόνο και με μικρή κατανάλωση χώρου, καθιστώντας τα κατάλληλα για τη μετάδοση σε περιβάλλοντα που απαιτούν γρήγορη ανταλλαγή δεδομένων, όπως στην παρακολούθηση σεισμών σε πραγματικό χρόνο.

6.2 Slarchive

Το Slarchive [13] αντιπροσωπεύει ένα σημαντικό βήμα προόδου στον τομέα της σεισμολογίας, καθώς αποτελεί ένα προηγμένο σύστημα αρχειοθέτησης και διαχείρισης σεισμικών δεδομένων. Αυτή η πλατφόρμα προσφέρει έναν ενιαίο χώρο αποθήκευσης για τεράστιους όγκους δεδομένων που προέρχονται από σεισμικές παρατηρήσεις, επιτρέποντας την ευκολότερη πρόσβαση, αναζήτηση και ανάκτηση πληροφοριών. Η δυνατότητα αποθήκευσης δεδομένων σε μορφές όπως τα SAC, MiniSEED και πολλές άλλες, καθιστά το slarchive πολύ ευέλικτο και προσαρμόσιμο στις ανάγκες της σεισμολογικής κοινότητας.

Ένα από τα κύρια πλεονεκτήματα του Slarchive είναι η δυνατότητά του να διατηρεί τις χρονικές σειρές δεδομένων με ακρίβεια και αξιοπιστία, ενώ παράλληλα παρέχει προηγμένες λειτουργίες αναζήτησης και φιλτραρίσματος. Η διασφάλιση της ακεραιότητας των δεδομένων και η δυνατότητα αναγνώρισης και διαχείρισης μεταδεδομένων προσθέτουν επιπλέον αξία στην πλατφόρμα. Επιπλέον, η δυνατότητα ενσωμάτωσης τεχνολογιών τεχνητής νοημοσύνης και μηχανικής μάθησης στο SLArchive επιτρέπει την αυτόματη αναγνώριση προτύπων και την εξαγωγή ενδιαφέρουσων πληροφοριών από τα σεισμικά δεδομένα.

Συνολικά, το Slarchive αντιπροσωπεύει έναν τεχνολογικά προηγμένο και συνολικά ολοκληρωμένο πόρο για την σεισμολογική κοινότητα, προσφέροντας τη δυνατότητα προηγμένης ανάλυσης και ερμηνείας σεισμικών δεδομένων, προωθώντας την επιστημονική έρευνα και συνεισφέροντας στην προαγωγή της κατανόησης των σεισμικών φαινομένων και της γεωδυναμικής εξέλιξης του πλανήτη.

Το slarchive συνδέεται σε έναν διακομιστή SeedLink, ζητά ροές δεδομένων και γράφει τα πακέτα που λαμβάνει σε δομές καταλόγων/αρχείων. Η ακριβής διάταξη των καταλόγων και των αρχείων ορίζεται σε μια συμβολοσειρά μορφής.

Οι υλοποιημένες διατάξεις είναι οι εξής:

- **SDS**: Η default δομή δεδομένων SeisComP
- **BUD**: Buffer of Uniform Data structure
- **DLOG**: Η παλιά δομή SeisComP/datalog για συμβατότητα προς τα πίσω (backwards compatibility)

Η διάρκεια για την οποία τα δεδομένα διατηρούνται στο αρχείο ελέγχεται από την δεσμευμένη παράμετρο **keep**. Το ίδιο το slarchive δεν καθαρίζει το αρχείο.

6.2.1 Ορισμός SDS

Το SDS είναι η βασική διάταξη καταλόγων και αρχείων που ακολουθείται στο SeisComP για αρχεία κυματομορφών. Ο βασικός κατάλογος του αρχείου ορίζεται από το archive. Η διάταξη SDS ορίζεται ως εξής (όπου <SDSdir> ο φάκελος archive):

```
<SDSdir>
  + year
    + network code
      + station code
        + channel code
          + one file per day and location, e.g. NET.STA.LOC.CHAN.D.YEAR.DOY
```

Εικόνα 25: *Archive Folder Structure [13]*

Παράδειγμα διαδρομής αρχείου (file path):

archive/Year/NET/STA/CHAN.TYPE/NET.STA.LOC.CHAN.TYPE.YEAR.DAY

Στον παρακάτω πίνακα παρουσιάζονται αναλυτικά τα επιμέρους μέρη που αποτελούν τη διαδρομή ενός αρχείου.

Σημειώνεται ότι οι λοιπές πληροφορίες που φαίνονται στον παρακάτω πίνακα εντοπίζονται και στο header ενός MiniSEED- όπως αναφέρεται στο 6.1- μορφή με την οποία αποθηκεύονται τα σεισμικά δεδομένα.

Field	Description
SDSdir	Arbitrary base directory
YEAR	4 digit YEAR
NET	Network code/identifier, 1-8 characters, no spaces
STA	Station code/identifier, 1-8 characters, no spaces
CHAN	Channel code/identifier, 1-8 characters, no spaces
TYPE	1 character, indicating the data type, provided types are: **D** Waveform data **E** Detection data **L** Log data **T** Timing data **C** Calibration data **R** Response data **O** Opaque data
LOC	Location identifier, 1-8 characters, no spaces
DAY	3 digit day of year, padded with zeros

Πίνακας 3: *Description of SDS Layout*

6.3 Ringserver

Ο όρος "ringserver" [24] αναφέρεται σε ένα συγκεκριμένο κομμάτι λογισμικού που χρησιμοποιείται στο πλαίσιο του SeedLink. Πιο συγκεκριμένα, ο ringserver είναι υπεύθυνος για τη διανομή πραγματικού χρόνου (real-time) δεδομένων σεισμικής δραστηριότητας σε δίκτυα παρατηρητηρίων. Λειτουργεί ως ένας διακομιστής που συλλέγει δεδομένα από διάφορους σταθμούς, επεξεργάζεται τα δεδομένα αυτά και τα διανέμει σε άλλες εφαρμογές ή δίκτυα που χρησιμοποιούν τα σεισμικά δεδομένα.

Ο ringserver λειτουργεί πάνω σε πρωτόκολλα επικοινωνίας SeedLink, τα οποία έχουν σχεδιαστεί για να είναι αποδοτικά και αξιόπιστα κατά τη μεταφορά σεισμικών δεδομένων. Ο όρος "ring" αναφέρεται στην κυκλική δομή- ring buffer ή αλλιώς κυκλικός buffer- που χρησιμοποιείται για τη διαχείριση των δεδομένων, επιτρέποντας στο σύστημα να διατηρεί μια συνεχή ροή πραγματικού χρόνου.

Ειδικότερα, το ring buffer είναι μια δομή δεδομένων που χρησιμοποιείται στον ringserver για τη διαχείριση των σεισμικών δεδομένων. Είναι ένας τύπος buffer που αποθηκεύει δεδομένα σε έναν κυκλικό πίνακα με σταθερό μέγεθος.

Η ιδέα πίσω από τον κυκλικό buffer είναι ότι, αντί να αυξάνεται δυναμικά το μέγεθος του buffer με την πάροδο του χρόνου, τα νέα δεδομένα αντικαθιστούν τα παλαιότερα δεδομένα όταν ο buffer γεμίσει. Αυτό δημιουργεί έναν κύκλο, εκείνο το οποίο έχει οδηγήσει στην ονομασία "κυκλικός buffer" ή "ring buffer".

Κατά τη λειτουργία του ringserver, τα νέα σεισμικά δεδομένα συνεχώς προστίθενται στον κυκλικό buffer. Όταν ο buffer γεμίσει, τα παλαιότερα δεδομένα αντικαθίστανται από τα νέα. Αυτό εξασφαλίζει ότι ο buffer διατηρεί πάντα ένα σταθερό μέγεθος, ενώ παράλληλα διατηρεί τη συνεχή ροή των πραγματικού χρόνου δεδομένων.

Η χρήση κυκλικού buffer είναι σημαντική στον τομέα της σεισμολογίας και της παρακολούθησης σεισμικής δραστηριότητας, καθώς επιτρέπει τη συνεχή καταγραφή και διανομή των σεισμικών δεδομένων, ακόμα και όταν προκύπτει μεγάλος όγκος δεδομένων.

6.4 SeisComP

Το SeisComP [29] πιθανότατα είναι το πιο διανεμημένο πακέτο λογισμικού για την παρακολούθηση σε πραγματικό χρόνο των σεισμών αλλά και άλλων σεισμικών γεγονότων. Παρέχει αυτόματη και διαδραστική απόκτηση σεισμολογικών δεδομένων, επεξεργασία και ανταλλαγή δεδομένων μέσω του διαδικτύου. Περιλαμβάνει διάφορα εργαλεία, μεταξύ των οποίων είναι και τα SeedLink και Slarchive.

Το πρωτόκολλο μετάδοσης δεδομένων SeedLink έχει γίνει ένα είδος παγκόσμιου προτύπου αφού έχει υιοθετηθεί από όλα τα σεισμικά κέντρα.

Κάποια από τα χαρακτηριστικά του SeisComP είναι τα εξής:

- Ισχυρή και αξιόπιστη αυτόματη επεξεργασία δεδομένων σε πραγματικό χρόνο ή κατά τη μεταεπεξεργασία.
- Φιλικές προς τον χρήστη γραφικές διεπαφές.
- Σύγχρονο και καλά συντηρημένο λογισμικό ανοικτού κώδικα στο GitHub [28], στο οποίο μπορεί όποιος επιθυμεί να γράψει κώδικα και να προτείνει βελτιώσεις πάνω στο υπάρχον σύστημα.

Η αρχική ανάπτυξη του SeisComP, όπως το γνωρίζουμε σήμερα, ξεκίνησε πριν σχεδόν δύο δεκαετίες με την προσπάθεια εξέλιξης στο GFZ (Γεωφυσικό Κέντρο Πότσνταμ) για τη δημιουργία προσθηκών σε ψηφιακούς μετατροπείς. Σήμερα, η ανάπτυξη και συντήρηση του SeisComP προωθείται από τις εταιρείες gempa GmbH και το GFZ.

6.5 Docker

Το Docker [14] αντιπροσωπεύει μια πλατφόρμα ανοιχτού κώδικα που επιτρέπει την εκτέλεση εφαρμογών και απλοποιεί τη διαδικασία ανάπτυξης. Οι εφαρμογές που ενσωματώνονται στο Docker συσκευάζονται ως "containers," περιλαμβάνοντας όλες τις απαραίτητες εξαρτήσεις. Αυτά τα containers λειτουργούν απομονωμένα πάνω από τον πυρήνα του λειτουργικού συστήματος. Παρά την ύπαρξη παρόμοιων τεχνολογιών για περισσότερο από 10 χρόνια, το Docker ξεχωρίζει ως καινοτόμα και ελπιδοφόρα λύση, προσφέροντας νέες δυνατότητες που οι προηγούμενες τεχνολογίες δεν παρείχαν.

Καταρχάς, παρέχει τη δυνατότητα δημιουργίας και διαχείρισης των containers. Οι προγραμματιστές μπορούν να συσκευάζουν τις εφαρμογές τους σε ελαφριά Docker containers. Αυτά τα εικονικοποιημένα περιβάλλοντα εφαρμογών μπορούν να εκτελούνται οπουδήποτε χωρίς τροποποίηση. Επιπλέον, το Docker μπορεί να μεταφέρει πολλά περισσότερα εικονικά περιβάλλοντα από ό,τι οι προηγούμενες τεχνολογίες, προσφέροντας ευελιξία και αποτελεσματικότητα.

Ακόμα, το Docker επιτρέπει τον εύκολο συντονισμό με τρίτα εργαλεία, επιτρέποντας άνετη ανάπτυξη και διαχείριση των Docker containers.

Το χαρακτηριστικό της αυτοματοποίησης των εφαρμογών κατά την ανάπτυξη σε container αποτελεί ένα ιδιαίτερα σημαντικό πλεονέκτημα του Docker. Σε ένα περιβάλλον container, όπου οι εφαρμογές είναι εικονικοποιημένες και εκτελούνται, το Docker προσθέτει ένα επιπλέον επίπεδο ανάπτυξης.

Σχεδιάζεται για να παρέχει γρήγορο και ελαφρύ περιβάλλον, επιτρέποντας αποτελεσματική λειτουργία του κώδικα και παρέχοντας ευκολία στον έλεγχο του κώδικα πριν την παραγωγή.

Τα βασικά στοιχεία του Docker περιλαμβάνουν:

- **Docker Daemon**: Πρόκειται για μια επίπονη διαδικασία παρασκηνίου που διαχειρίζεται τις Docker εικόνες, τα containers, τα δίκτυα και τους όγκους αποθήκευσης. Το Docker daemon επεξεργάζεται συνεχώς αιτήματα για το Docker API και τα διαχειρίζεται.
- **Docker Engine REST API**: Πρόκειται για ένα API που χρησιμοποιείται από εφαρμογές για την αλληλεπίδραση με το Docker daemon και μπορεί να προσπελαστεί από έναν πελάτη HTTP.
- **Docker CLI**: Πρόκειται για έναν πελάτη διεπαφής γραμμής εντολών που επιτρέπει την αλληλεπίδραση με το Docker daemon. Απλοποιεί σημαντικά τη διαχείριση των containers και αποτελεί έναν από τους βασικούς λόγους που οι προγραμματιστές προτιμούν τη χρήση του Docker.

6.5.1 Αρχιτεκτονική Docker

Η αρχιτεκτονική του Docker βασίζεται σε ένα μοντέλο πελάτη-διακομιστή και περιλαμβάνει τα εξής βασικά στοιχεία:

- Docker Client
- Docker Host
- Docker Network
- Docker Storage
- Docker Registry/Hub

6.5.1.1 Docker Client

Ο Docker Client λειτουργεί ως διεπαφή μεταξύ των χρηστών και του συστήματος Docker, επιτρέποντας την αλληλεπίδραση με το Docker. Ο Docker Client μπορεί να βρίσκεται είτε στον ίδιο εξυπηρετητή με τον daemon, είτε να συνδέεται με έναν απομακρυσμένο daemon σε έναν άλλο εξυπηρετητή. Ένας Docker Client μπορεί ακόμα να επικοινωνεί με περισσότερους από ένα daemon, παρέχοντας έτσι ευελιξία.

Η κύρια λειτουργία του Docker Client είναι να παρέχει ένα διαδραστικό Command Line Interface (CLI) που επιτρέπει στον χρήστη να εκδίδει εντολές για τη δημιουργία, εκτέλεση και διακοπή των Docker containers στον Docker daemon. Επιπλέον, διαδραματίζει σημαντικό ρόλο στη δρομολόγηση της

ανάκτησης εικόνων από το Docker Registry/Hub και στην εκτέλεσή τους σε έναν κατάλληλο Docker Host. Κατ' ουσίαν, ο Docker Client αποτελεί το εργαλείο με το οποίο ο χρήστης διαχειρίζεται και επικοινωνεί με το Docker σύστημα.

6.5.1.2 Docker Host

Ο εξυπηρετητής Docker παρέχει ένα πλήρες περιβάλλον για την εκτέλεση εφαρμογών, αποτελούμενο από τον daemon Docker, εικόνες, containers, δίκτυα και αποθήκευση. Ο daemon είναι υπεύθυνος για όλες τις ενέργειες που σχετίζονται με τα containers και λαμβάνει εντολές μέσω του Command Line Interface (CLI) ή του REST API. Επιπλέον, μπορεί να επικοινωνεί με άλλους daemon για τη διαχείριση των υπηρεσιών του.

Ο daemon ανακτά και δημιουργεί εικόνες και containers όπως ζητά ο πελάτης. Όταν ανακτά μια εικόνα, δημιουργεί ένα μοντέλο εργασίας για το container χρησιμοποιώντας ένα σύνολο εντολών γνωστό ως αρχείο κατασκευής. Αυτό το αρχείο κατασκευής μπορεί να περιλαμβάνει οδηγίες για τον daemon, φόρτωση προαπαιτούμενων στοιχείων πριν από την εκτέλεση του container, ή οδηγίες που εκτελούνται στην τοπική γραμμή εντολών αφού κατασκευαστεί το container.

6.5.1.3 Docker Network

Στη διαδικασία ανάπτυξης μιας εφαρμογής, τα βασικά αντικείμενα Docker που χρησιμοποιούνται περιλαμβάνουν τα εξής:

1. **Εικόνες (Images)**: Οι εικόνες αποτελούν ένα δυαδικό πρότυπο μόνο για ανάγνωση, το οποίο χρησιμοποιείται για τη δημιουργία container. Περιέχουν μεταδεδομένα που περιγράφουν τις δυνατότητες και τις ανάγκες του container, καθώς και πληροφορίες για την αποθήκευση και αποστολή εφαρμογών. Οι εικόνες επιτρέπουν τη συνεργασία μεταξύ προγραμματιστών και μπορούν να κοινοποιηθούν μέσω ιδιωτικών ή δημόσιων μητρώων όπως το Docker Hub.
2. **Containers:** Τα containers είναι περιβάλλοντα σε κάψουλα στα οποία εκτελούνται εφαρμογές. Καθορίζονται από την εικόνα και επιπλέον ρυθμίσεις κατά την εκκίνηση τους, περιλαμβανομένων συνδέσεων δικτύου και επιλογών αποθήκευσης. Τα containers έχουν πρόσβαση μόνο σε πόρους που ορίζονται στην εικόνα, προσφέροντας απομόνωση και ασφάλεια.
3. **Δικτύωση**: Το Docker παρέχει δικτυακές επιλογές που βασίζονται σε εφαρμογές και παρέχει διάφορες επιλογές. Υπάρχουν το προεπιλεγμένο δίκτυο Docker και τα δίκτυα που ορίζονται από τον χρήστη. Τα δίκτυα αυτά επιτρέπουν την ανάπτυξη ευέλικτων εφαρμογών και προσφέρουν απομόνωση και ανακάλυψη υπηρεσιών.

Αυτά τα αντικείμενα συνθέτουν το οικοσύστημα του Docker, παρέχοντας ένα ευέλικτο και απομονωμένο περιβάλλον για την ανάπτυξη και εκτέλεση εφαρμογών.

Άλλοι τύποι δικτύων αποτελούν τα δίκτυα που ορίζονται από τον χρήστη στο Docker και προσφέρουν επιπλέον ευελιξία και προσαρμογή στις ανάγκες της εφαρμογής. Πιο συγκεκριμένα:

1. **Δίκτυο Γέφυρας (Bridge Network)**: Παρόμοιο με το προεπιλεγμένο δίκτυο γέφυρας, αλλά διαφοροποιείται στο ότι δεν απαιτεί προώθηση θυρών για την επικοινωνία μεταξύ των containers εντός του δικτύου. Επίσης, παρέχει πλήρη υποστήριξη για αυτόματη ανακάλυψη δικτύου.
2. **Δίκτυο Επικάλυψης (Overlay Network)**: Χρησιμοποιείται όταν τα containers πρέπει να βρίσκονται σε ξεχωριστούς εξυπηρετητές για την επικοινωνία μεταξύ τους, όπως σε ένα κατανεμημένο δίκτυο. Η λειτουργία σμήνους πρέπει να είναι ενεργοποιημένη για ένα σύμπλεγμα Docker Engine για να ενταχθεί στην ίδια ομάδα.
3. **Δίκτυο Macvlan**: Χρησιμοποιείται για να επιτρέπει την έκθεση πόρων containers σε εξωτερικά δίκτυα χωρίς την ανάγκη προώθησης θυρών. Αυτό επιτυγχάνεται μέσω της χρήσης διευθύνσεων MAC αντί για διευθύνσεις IP.

Με αυτούς τους τρεις τύπους δικτύων, οι διαχειριστές μπορούν να προσαρμόσουν το περιβάλλον δικτύου σύμφωνα με τις απαιτήσεις της εφαρμογής και να διαμορφώσουν πολλαπλά δίκτυα για διάφορες ανάγκες.

6.5.1.4 Docker Storage

Οι επιλογές αποθήκευσης στο Docker προσφέρουν ευελιξία και δυνατότητα προσαρμογής στις ανάγκες της εκάστοτε εφαρμογής. Προσφέρονται οι παρακάτω τέσσερις επιλογές.

- **Τόμοι Δεδομένων (Volumes)**: Είναι η προτιμώμενη μέθοδος για μόνιμη αποθήκευση. Οι τόμοι δεδομένων είναι ανεξάρτητοι από τα containers και τα δεδομένα που αποθηκεύονται σε αυτούς είναι διαθέσιμα ακόμη και μετά την διακοπή ή διαγραφή του container. Είναι δυνατή η δημιουργία τόμων, ο διαμοιρασμός τους με πολλά containers και η εύκολη διαχείρισή τους.
- **Container Τόμοι Δεδομένων (Data Volume Containers**): Αυτή η μέθοδος επιτρέπει τη δημιουργία ενός εξειδικευμένου container που φιλοξενεί έναν τόμο και μπορεί να συνδεθεί με άλλα containers. Είναι μια εναλλακτική προσέγγιση για περιπτώσεις που απαιτείται ένα αποκλειστικό container για τη διαχείριση του τόμου.
- **Προσάρτηση Ευρετηρίου (Host Directory Mount)**: Αποτελεί έναν τόμο σε έναν κατάλογο στον εξυπηρετητή. Ο τόμος είναι προσαρτημένος απευθείας από τον εξυπηρετητή, και τα

δεδομένα αποθηκεύονται εκεί. Αυτή η μέθοδος παρέχει αποδοτικότητα και ταυτόχρονα μόνιμη αποθήκευση.

- **Προσθήκες Αποθήκευσης (Storage Drivers)**: Παρέχουν τη δυνατότητα για συσχέτιση του χώρου αποθήκευσης από τον εξυπηρετητή με μια εξωτερική πηγή, όπως μια συστοιχία αποθήκευσης ή μια συσκευή.

6.5.1.5 Docker Registry/Hub

Ένα μητρώο Docker (Docker Registry) λειτουργεί ως κεντρική τοποθεσία όπου μπορούν να αποθηκευτούν και να διαμοιραστούν εικόνες Docker. Αυτό το μητρώο μπορεί να είναι δημόσιο ή ιδιωτικό, ανάλογα με τις ανάγκες και τις προτιμήσεις των χρηστών.

Τα δημόσια μητρώα παρέχουν πρόσβαση σε ευρεία κοινότητα προγραμματιστών και χρηστών Docker. Δύο δημόσια μητρώα που αναφέρθηκαν είναι το Docker Hub και το Docker Cloud. Το Docker Hub είναι ένα από τα πιο δημοφιλή δημόσια μητρώα και περιλαμβάνει χιλιάδες διαθέσιμες εικόνες Docker που καλύπτουν πολλές εφαρμογές και περιπτώσεις χρήσης.

Τα ιδιωτικά μητρώα είναι χρήσιμα όταν οι οργανισμοί ή οι χρήστες επιθυμούν να διατηρήσουν έλεγχο και ασφάλεια στις εικόνες Docker τους. Με ένα ιδιωτικό μητρώο, οι χρήστες μπορούν να αποθηκεύουν και να διαμοιράζονται εικόνες Docker εντός του οργανισμού τους χωρίς να είναι προσβάσιμες από το ευρύ κοινό.

6.6 Python

Η Python [16] έχει κερδίσει μεγάλη δημοτικότητα για πολλούς λόγους, όπως είναι η αναγνωσιμότητα του κώδικα, η ευκολία χρήσης της, η ποικιλομορφία στις τεχνικές προγραμματισμού που υποστηρίζει, και η εκτεταμένη βιβλιοθήκη που παρέχει. Παρακάτω επισημαίνονται τα πλεονεκτήματά της:

1. **Αναγνωσιμότητα Κώδικα**: Η συντακτική απλότητα και η έμφαση στην αναγνωσιμότητα καθιστούν την Python φιλική προς τον προγραμματιστή. Η φιλοσοφία της Python ("The Zen of Python" [15]) προωθεί τη σαφήνεια και την απλότητα.
2. **Πολλαπλές Τεχνικές Προγραμματισμού**: Η υποστήριξη πολλαπλών παραδειγμάτων προγραμματισμού, όπως ο αντικειμενοστραφής και ο διαδικαστικός προγραμματισμός, επιτρέπει στους προγραμματιστές να επιλέξουν τον τρόπο προγραμματισμού που ταιριάζει καλύτερα στο έργο τους.
3. **Μεγάλη Βιβλιοθήκη**: Η πληθώρα των διαθέσιμων βιβλιοθηκών και πακέτων καθιστά ευκολότερη την ανάπτυξη εφαρμογών σε πολλούς τομείς, όπως οι επιστημονικές υπολογιστικές

εργασίες, οι διαδικτυακές εφαρμογές, και ο χειρισμός δεδομένων. **Επιστημονική Πληροφορική**: Οι εξειδικευμένες βιβλιοθήκες όπως NumPy, SciPy και Matplotlib καθιστούν την Python δημοφιλή στον τομέα της επιστημονικής πληροφορικής και της ανάλυσης δεδομένων.

4. **Στήριξη Διαδικτυακών Εφαρμογών**: Η ύπαρξη πλαισίων (frameworks) όπως το Django και το Flask επιτρέπει την εύκολη δημιουργία και συντήρηση διαδικτυακών εφαρμογών.

Συνολικά, η Python έχει εξελιχθεί σε μια πολυλειτουργική γλώσσα προγραμματισμού που καλύπτει ένα ευρύ φάσμα χρήσεων και εφαρμογών.

Στη συνέχεια περιγράφονται τα σημαντικότερα πακέτα που χρησιμοποιήθηκαν στην υλοποίηση της παρούσας διπλωματικής. Η έκδοση της Python που χρησιμοποιήθηκε είναι η 3.11.12.

6.6.1 pyOpenSSL

Η βιβλιοθήκη pyOpenSSL [17] είναι μια δέσμη εργαλείων που παρέχει δυνατότητες SSL/TLS για τη γλώσσα προγραμματισμού Python. Η συγκεκριμένη βιβλιοθήκη βασίζεται στην OpenSSL, μια προηγμένη βιβλιοθήκη κρυπτογραφίας, και παρέχει μια Pythonic διεπαφή για τη χρήση των λειτουργιών SSL/TLS.

Κύρια χαρακτηριστικά της pyOpenSSL:

1. **Κρυπτογραφία SSL/TLS**: Η βιβλιοθήκη επιτρέπει την ανάπτυξη ασφαλών εφαρμογών με τη χρήση του πρωτοκόλλου SSL/TLS. Αυτό είναι χρήσιμο για την ασφαλή μετάδοση δεδομένων μεταξύ διακομιστών και πελατών.
2. **Χρήση Πιστοποιητικών**: Η pyOpenSSL υποστηρίζει τη χρήση ψηφιακών πιστοποιητικών, τα οποία επιτρέπουν την αυθεντικοποίηση του server και, εάν απαιτείται, του client. Αυτό είναι σημαντικό για τη δημιουργία ασφαλούς επικοινωνίας.
3. **Διαχείριση Κλειδιών**: Η βιβλιοθήκη επιτρέπει τη διαχείριση και τη χρήση κρυπτογραφικών κλειδιών που απαιτούνται για την SSL/TLS επικοινωνία.

Στην υλοποίηση της παρούσας διπλωματικής η βιβλιοθήκη pyOpenSSl χρησιμοποιήθηκε για τη δημιουργία μιας SSL επικοινωνίας μεταξύ server- client. Παρακάτω αναφέρονται τα βασικά βήματα μιας τέτοιας επικοινωνίας:

1. **Δημιουργία Κλειδιών και Πιστοποιητικών**: Πριν από την αρχή της επικοινωνίας, ο server χρειάζεται ένα κρυπτογραφικό κλειδί και ένα ψηφιακό πιστοποιητικό. Η pyOpenSSL βοηθά στη δημιουργία και τη διαχείριση αυτών των κλειδιών.

2. **Δημιουργία SSL/TLS Context**: Και ο server και ο client πρέπει να δημιουργήσουν ένα SSL/TLS context, το οποίο καθορίζει τις ρυθμίσεις της ασφαλούς επικοινωνίας.
3. **Διαχείριση Συνεδρίας**: Η pyOpenSSL παρέχει εργαλεία για τη διαχείριση των συνεδριών SSL/TLS, περιλαμβανομένου του χειρισμού κρυπτογραφημένων δεδομένων.
4. **Ανταλλαγή Δεδομένων**: Με το SSL/TLS context παραμετροποιημένο, ο server και ο client μπορούν να ανταλλάξουν ασφαλή δεδομένα μέσω της pyOpenSSL.

Συνοψίζοντας, η pyOpenSSL είναι ένα ισχυρό εργαλείο που επιτρέπει στους προγραμματιστές Python να υλοποιήσουν ασφαλείς συνδέσεις SSL/TLS μεταξύ server και client, επιτρέποντας την ασφαλή μετάδοση δεδομένων.

6.6.2 ObsPy

Το ObsPy [18] είναι ένα εξαιρετικά χρήσιμο και ισχυρό εργαλείο για την επεξεργασία σεισμολογικών δεδομένων στο περιβάλλον της Python. Κάποια βασικά σημεία είναι τα εξής:

1. **Παροχή Βασικών Ρουτινών**: Η παροχή βασικών ρουτινών για την ανάγνωση, επεξεργασία και αποθήκευση σεισμολογικών δεδομένων, όπως SEED/MiniSEED, SAC και άλλα, καθιστά το ObsPy ευέλικτο για σεισμολόγους.
2. **Αντικειμενοστραφής Σχεδιασμός**: Το γεγονός ότι το ObsPy είναι πλήρως αντικειμενοστραφές επιτρέπει την αντιμετώπιση σεισμολογικών δεδομένων ως αντικείμενα, όπως το αντικείμενο "Trace," που απλοποιεί τη διαχείριση και την επεξεργασία. Πιο συγκεκριμένα το σεισμογραφικό ενός καναλιού αφαιρείται ως αντικείμενο «Trace». Ένα αντικείμενο Trace έχει κάποιο πυρήνα καθώς και κάποια απαιτούμενα χαρακτηριστικά όπως είναι το διάστημα δείγματος και ο αριθμός των δειγμάτων. Ένα λεξικό python χρησιμοποιείται για τη διατήρηση βοηθητικών μεταδεδομένων που αποθηκεύονται ως χαρακτηριστικά κεφαλίδας σε ένα πακέτο
3. **Ευρεία Υποστήριξη Προτύπων**: Η υποστήριξη για πολλά πρότυπα αρχείων όπως SEED/MiniSEED, SAC, XML-SEED και πρωτόκολλα ανάκτησης δεδομένων όπως ArcLink (πρωτόκολλο αίτησης κατανεμημένων δεδομένων για πρόσβαση σε αρχειοθετημένα δεδομένα κυματομορφής) επιτρέπει τη συνεργασία με διάφορα σεισμολογικά συστήματα.
4. **Εκτεταμένη Χρήση Βοηθητικών Αντικειμένων**: Η χρήση βοηθητικών αντικειμένων όπως το UTCDateTime, Catalog, και Inventory καθιστά ευκολότερη τη διαχείριση μεταδεδομένων και τη συνολική επεξεργασία.
    - **UTCDateTime**- χρησιμοποιείται για τη διατήρηση απόλυτα χρονικών σημάνσεων.
    - **Catalog**- Container που χρησιμοποιείται για τη διατήρηση πληροφοριών προέλευσης.

- **Inventory**- Container για εσωτερικό χειρισμό μεταδεδομένων σταθμού.

5. **Υποστήριξη RESTful API**: Η δυνατότητα ανάκτησης δεδομένων μέσω RESTful API επιτρέπει την αποτελεσματική διανομή σεισμολογικών δεδομένων.

Το ObsPy αποτελεί ένα σημαντικό εργαλείο για τη σεισμολογική κοινότητα, και η χρήση της Python του παρέχει την ευελιξία και τη δύναμη της γλώσσας για αποδοτική ανάπτυξη και χρήση.

6.6.3 Secrets

Η βιβλιοθήκη secrets [19] της Python παρέχει λειτουργίες για τη δημιουργία ασφαλών τυχαίων δεδομένων, όπως κλειδιά και τυχαίους αριθμούς, χωρίς να αποθηκεύονται μόνιμα στη μνήμη. Είναι χρήσιμη όταν χρειάζονται τυχαία δεδομένα. τα οποία όμως πρέπει να παραχθούν με ασφαλή τρόπο. Τέτοια παραδείγματα αποτελούν η δημιουργία κλειδιών αυθεντικοποίησης (API keys), cookies, κρυπτογραφικών κλειδιών και άλλων.

Ένα παράδειγμα αποτελεί η συνάρτηση **secrets.token_bytes(n)**, η οποία επιστρέφει n τυχαία bytes που παράγονται με αποδεκτό τρόπο για κρυπτογραφικές εφαρμογές. Γενικά, οι συναρτήσεις της βιβλιοθήκης secrets παρέχουν τρόπους για να δημιουργούνται τυχαία, ασφαλή δεδομένα χρησιμοποιώντας κρυπτογραφικά ασφαλείς μεθόδους. Είναι σημαντικό να χρησιμοποιείται η βιβλιοθήκη secrets για τέτοιου είδους εργασίες, αφού οι απλές συναρτήσεις τυχαίων αριθμών της Python (όπως η random) δεν είναι κατάλληλες για κρυπτογραφική χρήση λόγω της προβλεψιμότητάς τους.

6.6.4 Crypto

Η βιβλιοθήκη Crypto [20] της Python αντιπροσωπεύει ένα εκτεταμένο εργαλείο για τη διαχείριση κρυπτογραφικών λειτουργιών στη γλώσσα προγραμματισμού Python. Σχεδιασμένη για να παρέχει υψηλή ασφάλεια και απόδοση, η βιβλιοθήκη υποστηρίζει μια εκτεταμένη γκάμα αλγορίθμων κρυπτογραφίας, συμπεριλαμβανομένων των δημοφιλών AES, DES, και Blowfish. Αυτό επιτρέπει στους προγραμματιστές να επιλέγουν τον κατάλληλο αλγόριθμο για τις συγκεκριμένες ανάγκες ασφαλείας.

Εκτός από τους αλγορίθμους κρυπτογραφίας, η βιβλιοθήκη παρέχει επίσης δυνατότητες κατακερματισμού με διάφορους αλγορίθμους hash, όπως το MD5 και το SHA. Οι παραπάνω δυνατότητες είναι χρήσιμες όσον αφορά στον έλεγχο ακεραιότητας δεδομένων και τη δημιουργία αναγνωριστικών για τα δεδομένα αυτά.

Επιπλέον, η βιβλιοθήκη προσφέρει υποστήριξη για τη δημιουργία ψηφιακών υπογραφών και τη διαχείριση πιστοποιητικών, γεγονός ιδιαίτερα κρίσιμο για την ασφαλή επικοινωνία και αναγνώριση της αυθεντικότητας των δεδομένων.

Γενικότερα, η βιβλιοθήκη Crypto αποτελεί κεντρικό εργαλείο για τους προγραμματιστές που επιδιώκουν υψηλή ασφάλεια και ποικιλομορφία λειτουργιών κρυπτογράφησης στην Python. Η συνεχής της ανάπτυξη και συντήρηση την καθιστούν ιδανική για εφαρμογές που απαιτούν αξιόπιστη προστασία δεδομένων και ασφαλή επικοινωνία.

6.6.5 Watchdog

Η βιβλιοθήκη watchdog [30] χρησιμοποιείται για την παρακολούθηση αλλαγών σε αρχεία και φακέλους, προσφέροντας έναν εύκολο και ευέλικτο τρόπο για αντίδραση σε δυναμικές αλλαγές σε ένα σύστημα αρχείων.

Ο προγραμματιστής δημιουργεί έναν παρατηρητή (Observer) που παρακολουθεί συγκεκριμένους φακέλους ή αρχεία. Ο παρατηρητής αυτός χρησιμοποιεί έναν χειριστή συμβάντων (FileSystemEventHandler) που προσαρμόζεται ανάλογα με τις ανάγκες του προγραμματιστή. Σε κάθε τροποποίηση, δημιουργία ή διαγραφή αρχείου, ο παρατηρητής ειδοποιεί την εφαρμογή.

Ένα παράδειγμα χρήσης περιλαμβάνει τη δημιουργία μιας υποκλάσης FileSystemEventHandler με μεθόδους όπως on_modified(), on_created(), κ.λπ., για να ανταποκρίνεται σε συγκεκριμένα σενάρια. Με τον τρόπο αυτό, η εφαρμογή μπορεί να προσαρμοστεί ανάλογα με τις ανάγκες του χρήστη, όπως η αυτόματη επαναφόρτωση σε αλλαγές πηγαίου κώδικα, η αυτόματη επαναφόρτωση εικόνων, κ.ά.

Το κυριότερο κομμάτι του κώδικα είναι η δημιουργία του Observer, ο οποίος ρυθμίζεται για τον συγκεκριμένο φάκελο παρακολούθησης. Κατά την εκκίνηση του Observer, η εφαρμογή εισέρχεται σε έναν βρόχο που παραμένει ανοικτός, επιτρέποντας την συνεχή παρακολούθηση. Ο Observer διακόπτεται κατά τη διακοπή του προγράμματος.

7

Περιγραφή Υλοποίησης

Στο παρόν κεφάλαιο, γίνεται λεπτομερή ανάλυση της υλοποίησης που εφαρμόστηκε στο πλαίσιο της διπλωματικής εργασίας. Συγκεκριμένα, στην ενότητα 7.1 εξετάζεται εκτενώς ο τρόπος με τον οποίο αποκτώνται και διαβιβάζονται τα σεισμικά δεδομένα σε έναν σεισμικό σταθμό. Στην ενότητα 7.2 παρουσιάζεται η νέα αρχιτεκτονική που υιοθετήθηκε, συνοδευόμενη από αναφορές στις αλλαγές που εφαρμόστηκαν για την επίτευξη της επιθυμητής ασφάλειας. Στη συνέχεια, η ενότητα 7.3 παρουσιάζει σύντομα τις δομές δεδομένων που χρησιμοποιούνται γενικά εκτενώς κατά τη διαδικασία μεταφοράς των σεισμικών δεδομένων. Τέλος, στην ενότητα 7.4 παρέχονται λεπτομερείς πληροφορίες σχετικά με τα διάφορα στοιχεία που αποτέλεσαν και συνέβαλαν στη νέα αρχιτεκτονική που δημιουργήθηκε.

Σημαντικό είναι να τονιστεί ότι η ενότητα 7.2 εστιάζει ειδικότερα στην ανάδειξη της ανανεωμένης δομής και των τεχνολογικών προσαρμογών που υιοθετήθηκαν για να διασφαλιστεί η ιδιαίτερα επιθυμητή ασφάλεια του συστήματος. Τα επιμέρους βήματα της υλοποίησης περιγράφονται διεξοδικά, παρέχοντας ένα πλήρες πλαίσιο κατανόησης για την εξέλιξη της διαδικασίας. Ειδικότερα, η ενότητα 7.4 παρουσιάζει εκτενώς τα διάφορα στοιχεία της νέας αρχιτεκτονικής, προσφέροντας μια σφαιρική κατανόηση των επιτευγμάτων και των προσθηκών που ενσωματώθηκαν στο έργο.

7.1 Τρέχουσα Αρχιτεκτονική Συστήματος SeedLink

Ένας σεισμογράφος γενικά αποθηκεύει δεδομένα σε μορφή MiniSEED συνεχώς στην κάρτα SD του, και μπορεί επίσης να εκπέμπει αυτά τα πακέτα 512-byte μέσω μιας σύνδεσης σειριακής σε άλλα συστήματα λήψης δεδομένων (Data Reception System) μέσω παραδείγματος χάριν RS232 ή ακόμη και ασύρματης σύνδεσης. Κάθε ένα από αυτά τα συστήματα περιλαμβάνει τη δυνατότητα να μοιραστεί

δεδομένα μέσω του SeedLink, συνήθως προς τον κεντρικό σεισμικό σταθμό που τα διαχειρίζεται και αποθηκεύει.

Το SEED αναφέρεται στο Πρότυπο σχετικά με την Ανταλλαγή Σεισμικών Δεδομένων (Standard for the Exchange of Earthquake Data), το οποίο σχεδιάστηκε τη δεκαετία του 1980 και αποτελεί πλέον ένα κοινό μορφότυπο για σεισμικά δεδομένα. Τα περισσότερα δεδομένα χρονοσειρών (κυματομορφές) χρησιμοποιούν το υποσύνολο MiniSEED (βλ. 6.1) του προτύπου SEED, το οποίο διαμορφώνει τα δεδομένα σε εγγραφές 512-byte για τη μεταφορά των δεδομένων, αλλά περιλαμβάνει περιορισμένες πληροφορίες σχετικά με αυτά δεδομένα. Τα αρχεία Dataless SEED αποτελούν το αντίστοιχο ως προς τις κυματομορφές MiniSEED. Περιλαμβάνουν τα μεταδεδομένα που περιγράφουν λεπτομερώς τα σεισμικά δεδομένα, συμπεριλαμβανομένων του τόπου εγγραφής των δεδομένων, του τύπου των οργάνων που χρησιμοποιήθηκαν για την εγγραφή των δεδομένων, και του τρόπου μετατροπής των δεδομένων από ακατέργαστους αριθμούς σε κίνηση του εδάφους ή άλλες μονάδες.

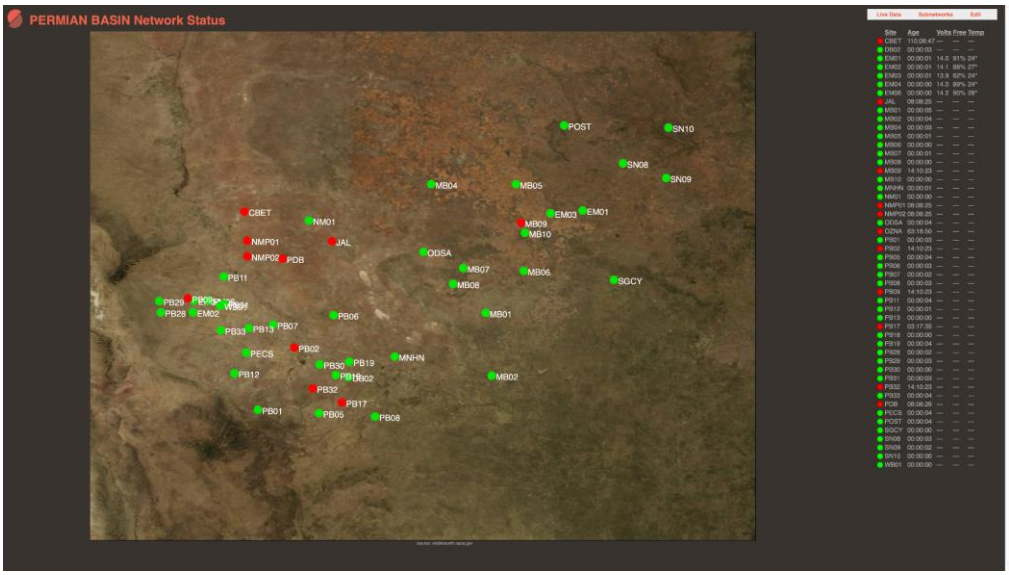
Το SeedLink (για περισσότερες λεπτομέρειες βλ. 6.1) είναι ένα ανθεκτικό πρωτόκολλο μετάδοσης δεδομένων που σχεδιάστηκε για τη διαβίβαση πακέτων MiniSEED στο Διαδίκτυο και άλλα δίκτυα TCP/IP. Ένας διακομιστής SeedLink αποτελεί πηγή δεδομένων, η οποία μπορεί να είναι μόνο ένας σταθμός, ένα δίκτυο σταθμών ή μια συλλογή δικτύων ή σταθμών, με κάθε σταθμό να μπορεί πιθανώς να μοιράζεται πολλά κανάλια δεδομένων.

Στην περίπτωση που χρειάζεται να διαμορφωθεί ένας διακομιστής SeedLink σε έναν απομακρυσμένο σεισμογράφο, αυτό μπορεί να γίνει με τη χρήση παραδείγματος χάριν ενός Raspberry Pi, στον οποίο θα έχει εγκατασταθεί ένα image λειτουργικού συστήματος που υποστηρίζει SeedLink. Τα δεδομένα που θα προέρχονται από τον σεισμογράφο θα είναι στη συνέχεια προσβάσιμα μέσω του παραπάνω διακομιστή ενώ ακόμη μπορούν να αποθηκευτούν σε ένα USB D rive που συνδέεται στο Raspberry Pi.

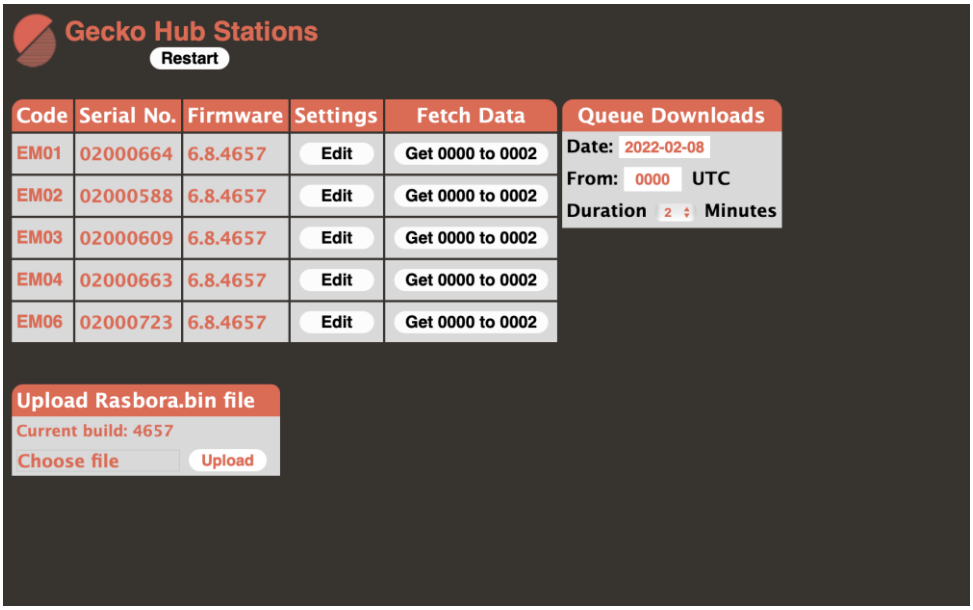
Αν στο τοπικό δίκτυο υπάρχουν πολλοί σεισμογράφοι, ο καθένας μπορεί να μεταδίδει δεδομένα σε έναν υπολογιστή μέσω ενός προσαρμογέα (adaptor) παραδείγματος χάριν RS232-to-Ethernet/WiFi (ή απευθείας μέσω καλωδίου USB).

Μια εναλλακτική λύση είναι να στέλνονται δεδομένα από έναν σεισμογράφο σε έναν eqServer που φιλοξενείται στο νέφος (cloud-hosted)- έναν υπολογιστή Ubuntu που λειτουργεί σε κάποιο σημείο του Διαδικτύου και είναι εγκατεστημένος με διάφορα εργαλεία, συμπεριλαμβανομένου ενός διακομιστή SeedLink για τη μετέπειτα μετάδοση δεδομένων (onward data distribution). Η παραπάνω σύνδεση μπορεί να υλοποιηθεί μέσω οποιασδήποτε διαθέσιμης σύνδεσης Διαδικτύου, όπως WiFi, Ethernet ή ακόμα και κυψελικών ή δορυφορικών συνδέσεων δεδομένων για απομακρυσμένους σταθμούς. Ο eqServer διαχειρίζεται μέσω ενός web browser και παρέχει επιπλέον δυνατότητες, όπως χάρτες (**Εικόνα 26**) και γραφήματα υγείας του σταθμού, εργαλεία αυτόματου εντοπισμού και ειδοποίησης για σεισμούς

(**Εικόνα 28**), καθώς επίσης και ιστοσελίδες διαχείρισης του σεισμογράφου από απόσταση (**Εικόνα 27**). Στις παρακάτω εικόνες τα παραδείγματα που παρουσιάζονται αφορούν τον σεισμογράφο Gecko [21].



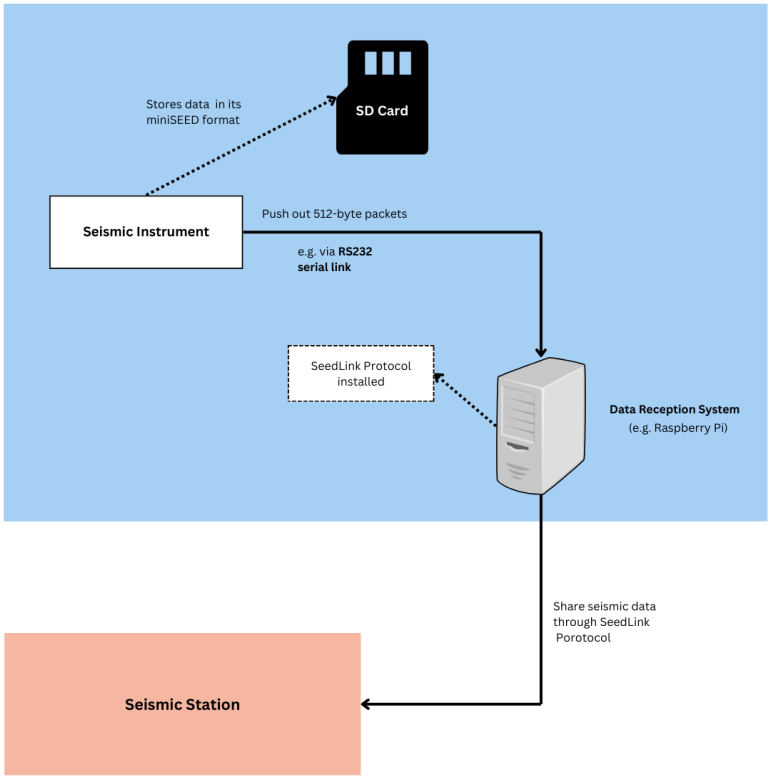
Εικόνα 26: *EqServer- Network Status [21]*



Εικόνα 27: *EqServer- Hub Stations [21]*

Εικόνα 28: *EqServer- History [21]*

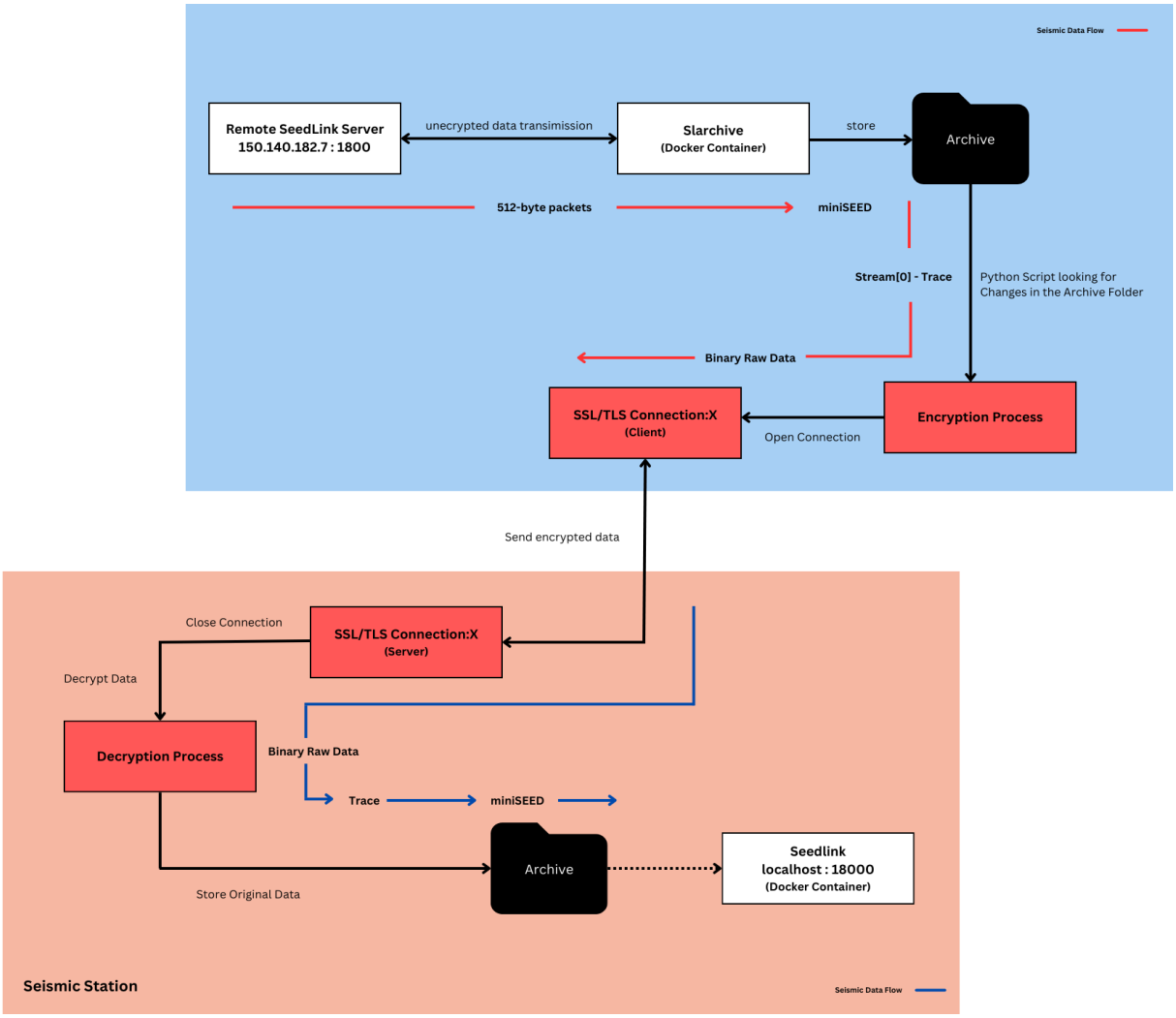
Στην **Εικόνα 29** παρουσιάζεται ένα χαρακτηριστικό σύστημα απόκτησης και μεταφοράς σεισμικών δεδομένων, χρησιμοποιώντας ένα σεισμογράφο και το πρωτόκολλο SeedLink.



Εικόνα 29: *Seismic Data Transmission via SeedLink Protocol*

7.2 Περιγραφή Νέας Αρχιτεκτονικής Συστήματος

Στο **Εικόνα 30**, παρουσιάζεται η νέα αρχιτεκτονική του συστήματος, η οποία βασίζεται στη δομή που αναφέρθηκε στην προηγούμενη ενότητα. Οι τροποποιήσεις που πραγματοποιήθηκαν, έγιναν με σκοπό την ομαλή ενσωμάτωση της κρυπτογραφικής διαδικασίας κατά τη μεταφορά των σεισμικών δεδομένων από το εκάστοτε Data Reception System προς έναν σεισμικό σταθμό. Επομένως, οι αλλαγές έγιναν πάνω στο γαλάζιο και πορτοκαλί τμήμα της **Εικόνας 29**.



Εικόνα 30: *Transmission of Seismic Data- New Architecture*

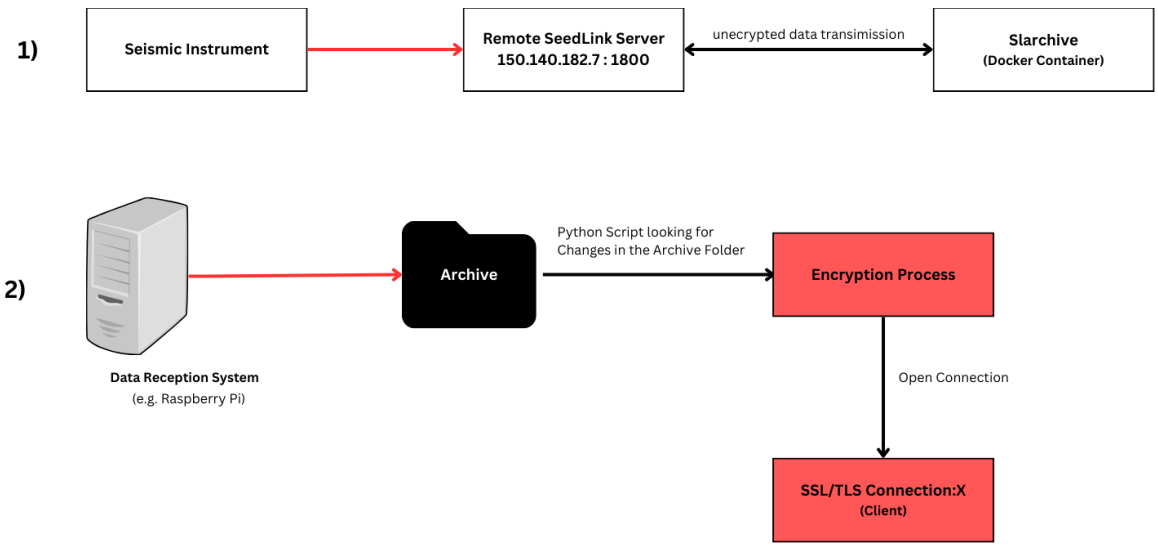
Το γαλάζιο τμήμα αναπαριστά τον σεισμογράφο που παράγει και αποστέλλει τις σεισμικές πληροφορίες στο Data Reception System. Δεδομένου ότι η χρήση πραγματικού σεισμογράφου δεν είναι εφικτή, ο τελευταίος αντικαταστάθηκε από το Slarchive (βλ. 6.2), προκειμένου να προσομοιαστεί το real-time data acquisition. Η SD Card του σεισμογράφου, όπου συνήθως αποθηκεύονται τα σεισμικά δεδομένα, αντικαταστάθηκε από έναν φάκελο "archive" για την αποθήκευση των αρχείων MiniSEED.

Τα τμήματα που εμφανίζονται με κόκκινο χρώμα αποτελούν νέες προσθήκες στο SeedLink σύστημα μεταφοράς σεισμικών δεδομένων. Προστέθηκαν με σκοπό την επίτευξη της κρυπτογράφησης/αποκρυπτογράφησης των σεισμικών δεδομένων και την ασφαλή διαβίβασή τους, σε αντίθεση με το προηγούμενο SeedLink σύστημα όπου αυτά τα στοιχεία δεν υπήρχαν.

Το πορτοκαλί τμήμα αντιστοιχεί στον κεντρικό server ενός σεισμικού σταθμού, ο οποίος αποθηκεύει τα δεδομένα που λαμβάνει. Στο σύστημα αυτό έχει εγκατασταθεί το πρωτόκολλο SeedLink (localhost) στο port 1800, ώστε να επαληθευτεί ότι, αφού τα δεδομένα σταλούν, αποκρυπτογραφηθούν και αποθηκευτούν στον server, δεν διαφοροποιείται η χρήση και η εφαρμογή των ήδη εγκατεστημένων εργαλείων πάνω σε αυτά (παραδείγματος χάριν SeedLink, FDSN Web Services [32]). Για το λόγο αυτό τα σεισμικά δεδομένα αποθηκεύονται στο φάκελο "archive", αφού όλα τα προαναφερθέντα εργαλεία αναζητούν τον συγκεκριμένο φάκελο προκειμένου να αποκτήσουν πρόσβαση σε αυτά και να τα επεξεργαστούν.

Τα κόκκινα βέλη που εντοπίζονται στο γαλάζιο τμήμα, καθώς και τα μπλε βέλη που εντοπίζονται στο πορτοκαλί τμήμα, αναφέρονται στη ροή και τη μορφή των δεδομένων που ακολουθούν σε κάθε επιμέρους στοιχείο της προτεινόμενης αρχιτεκτονικής κατά τη διάρκεια της μεταφοράς τους από τον σεισμογράφο έως τελικά στον κεντρικό σεισμικό σταθμό.

Στην **Εικόνα 31** παρουσιάζεται η αντιστοίχιση των στοιχείων της **Εικόνα 29** με τα στοιχεία της **Εικόνα 30**.



Εικόνα 31: *Typical SeedLink Server- New Proposed Architecture*

Επισημαίνεται πως κάθε επιμέρους στοιχείο της προτεινόμενης αρχιτεκτονικής που εντοπίζεται στην **Εικόνα 30,** θα αναλυθεί λεπτομερώς στην ενότητα 7.4.

7.3 Δομές και Ροή Σεισμικών Δεδομένων

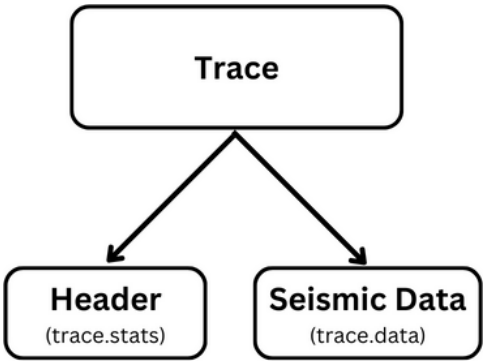
Στο πλαίσιο αυτής της ενότητας, ο στόχος είναι η κατανόηση και εξοικείωση με τις δομές δεδομένων που χρησιμοποιούνται για τη μεταφορά και αποθήκευση των σεισμικών δεδομένων.

Trace

Αναφέρεται σε ένα αντικείμενο που αντιπροσωπεύει μια χρονοσειρά παρατηρήσεων και πιο συγκεκριμένα σεισμικών δεδομένων. Κάθε "Trace" περιέχει πληροφορίες όπως ο τύπος των δεδομένων, οι χρονικές σημάνσεις, και οι ίδιες οι χρονοσειρές.

Η χρήση του "Trace" στην ObsPy είναι βασική για τη φόρτωση, επεξεργασία και ανάλυση σεισμικών δεδομένων. Χρησιμοποιείται ευρέως στο πλαίσιο εφαρμογών που αφορούν τη γεωφυσική, την επιστήμη του εδάφους, και άλλους τομείς που σχετίζονται με την παρακολούθηση των σεισμών και των γεωφυσικών φαινομένων.

Στην **Εικόνα 32** φαίνονται τα επιμέρους στοιχεία που αποτελούν μία δομή Trace.



Εικόνα 32: *Trace Architecture*

Το Header (trace.stats) ουσιαστικά περιέχει τα metadata ενός αντικειμένου Trace. Τα τελευταία είναι σημαντικά, καθώς παρέχουν απαραίτητες πληροφορίες για την προέλευση, τη χρονική σήμανση, τη μορφοποίηση και την ποιότητα των δεδομένων, συμβάλλοντας στην ακριβή επεξεργασία και ερμηνεία τους.

Παρακάτω περιγράφονται λεπτομερώς τα πεδία που χαρακτηρίζονται ως metadata και αποτελούν το Header ενός Trace.

1. **network (δίκτυο)**- Αναφέρεται στο σύστημα παρακολούθησης στο οποίο ανήκει το σταθμός.
2. **station (σταθμός)**- Αναφέρεται στον κωδικό του σταθμού παρακολούθησης.

3. **location (τοποθεσία)**- Αναφέρεται στην τοποθεσία του σταθμού. Αυτή η πληροφορία είναι πολύ σημαντική για την ακριβή κατανόηση της πηγής των σεισμικών δεδομένων.
4. **channel (κανάλι)**- Αναφέρεται στον τύπο του καναλιού παρακολούθησης.
5. **starttime (αρχή χρονικής σήμανσης)**- Η χρονική στιγμή έναρξης των σεισμικών δεδομένων.
6. **endtime (τέλος χρονικής σήμανσης)**- Η χρονική στιγμή λήξης των σεισμικών δεδομένων.
7. **sampling_rate (ρυθμός δειγματοληψίας)**- Ο ρυθμός με τον οποίο λαμβάνονται δείγματα ανά δευτερόλεπτο.
8. **delta (χρονικό διάστημα δειγματοληψίας)**- Το χρονικό διάστημα μεταξύ δειγμάτων.
9. **npts (αριθμός δειγμάτων)**- Ο συνολικός αριθμός των δειγμάτων.
10. **calib (βαθμονόμηση)**- Συντελεστής βαθμονόμησης που χρησιμοποιείται για τον υπολογισμό των πραγματικών τιμών της φυσικής μεταβολής.
11. **_format (μορφή)**- Η μορφή των σεισμικών δεδομένων, στην περίπτωση αυτή το "MSEED" (MiniSEED).
12. **mseed (MiniSEED metadata)**: Ένα αντικείμενο που περιλαμβάνει πρόσθετες πληροφορίες για τα MiniSEED δεδομένα, όπως η ποιότητα των δεδομένων, ο αριθμός των εγγραφών, η κωδικοποίηση κ.λπ.

Η **Εικόνα 33** αποτελεί παράδειγμα Header ενός Trace.

```
        network: HP
        station: UPR
       location:
        channel: HHN
      starttime: 2023-12-04T21:19:29.622500Z
        endtime: 2023-12-04T21:21:19.862500Z
  sampling_rate: 100.0
          delta: 0.01
           npts: 11025
          calib: 1.0
        _format: MSEED
          mseed: AttribDict({'dataquality': 'D', 'number_of_records': 40, 'encoding': 'STEIM2', 'byteorder': '>', 'record_length': 512, 'filesize': 20480})
```

Εικόνα 33: *Header of a Trace- Example*

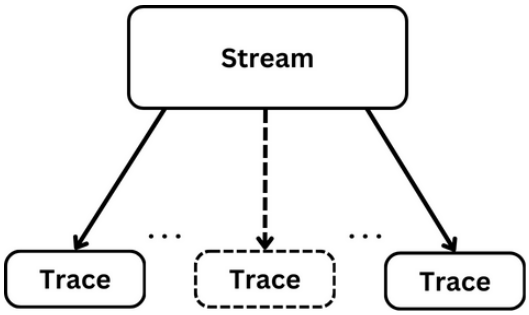
Το Seismic Data (trace.data) όπως είναι προφανές, περιέχει τα ίδια τα σεισμικά δεδομένα που έχουν καταγραφεί από τον σεισμογράφο. Αποτελεί ένα NumPy πίνακα που περιλαμβάνει τις χρονοσειρές των σεισμικών δεδομένων. Κάθε στοιχείο του πίνακα αντιπροσωπεύει την αμφιωτική κίνηση (amplitude) του εδάφους για ένα συγκεκριμένο χρονικό σημείο. Παράδειγμα τέτοιου πίνακα φαίνεται παρακάτω.

[-5074 -4845 -4821 ... -5975 -5722 -5555]

Stream

Το Stream είναι μια συλλογή από αντικείμενα Trace. Αντιπροσωπεύει ένα σύνολο σεισμικών καναλιών που συλλέγονται ταυτόχρονα από διάφορα σημεία. Ακόμη, επιτρέπει την εργασία με πολλαπλά κανάλια ταυτόχρονα, όπως η φόρτωση, η αποθήκευση, και η επεξεργασία δεδομένων σεισμικών καναλιών

Ουσιαστικά, ένα Stream είναι απλώς μια συλλογή από αντικείμενα Trace (**Εικόνα 34**) όπου κάθε Trace στο Stream αντιπροσωπεύει τα δεδομένα από ένα συγκεκριμένο κανάλι. Με αυτόν τον τρόπο, το Stream επιτρέπει τη διαχείριση και την ανάλυση συνόλων δεδομένων σεισμικών καναλιών.



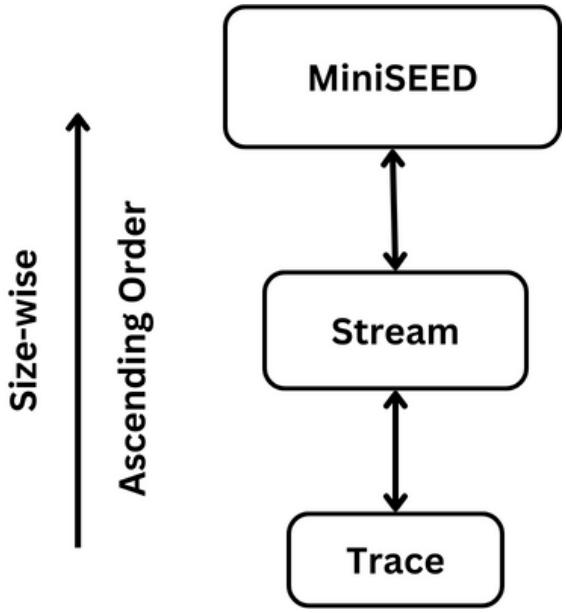
Εικόνα 34: *Stream*

MiniSEED

Είναι ένας μικρός, συμπιεσμένος τύπος δεδομένων που χρησιμοποιείται για την αποθήκευση και τη μεταφορά ψηφιακών εγγραφών από σεισμογράφους. Ορίζεται από τον Οργανισμό Ερευνών Γης (IRIS), έναν διεθνή οργανισμό που ασχολείται με τη συλλογή, τη διάθεση και την ανάλυση δεδομένων σεισμικής δραστηριότητας (για περισσότερες πληροφορίες βλ. 6.1).

Όταν διαβάζονται δεδομένα από ένα αρχείο MiniSEED μέσω της ObsPy, τα δεδομένα αυτά συνήθως φορτώνονται σε ένα αντικείμενο stream του ObsPy. Έτσι, το stream περιλαμβάνει τις πληροφορίες και τις επιλογές επεξεργασίας για ένα σύνολο χρονοσειρών σεισμικών δεδομένων, τα οποία μπορούν να προέρχονται από ένα ή περισσότερα αρχεία MiniSEED. Η χρήση του stream επιτρέπει στον χρήστη την εύκολη επεξεργασία, ανάλυση και αναπαράσταση των σεισμικών δεδομένων, χρησιμοποιώντας τα εργαλεία που παρέχει το ObsPy.

Στην **Εικόνα 36** παρουσιάζεται διαγραμματικά η ροή που ακολουθούν τα σεισμικά δεδομένα σύμφωνα με τις δομές δεδομένων που αναφέρθηκαν παραπάνω. Αρχικώς, τα δεδομένα αποθηκεύονται σε ένα αρχείο τύπου MiniSEED. Κατόπιν, με τη χρήση της βιβλιοθήκης ObsPy, διαβάζοντας τα δεδομένα, φορτώνονται σε ένα αντικείμενο τύπου "stream". Επιλέγοντας ένα συγκεκριμένο στοιχείο από το "stream", το επιλεγμένο αυτό δείγμα φορτώνεται σε ένα αντικείμενο τύπου "Trace".



Εικόνα 35: *Seismic Data Flow*

7.4 Περιγραφή Επιμέρους Στοιχείων της Προτεινόμενης Αρχιτεκτονικής

Στο πλαίσιο αυτού του κεφαλαίου, θα παρουσιαστούν εκτενώς τα επιμέρους στοιχεία που χρησιμοποιήθηκαν για την υλοποίηση της προτεινόμενης αρχιτεκτονικής.

7.4.1 SeedLink

Το πρωτόκολλο SeedLink υλοποιήθηκε μέσω Docker σε container, τρέχοντας το σύστημα τοπικά στον υπολογιστή (localhost). Για την ολοκληρωμένη υλοποίηση του SeedLink server, έχει ενσωματωθεί ο κώδικας που απαιτείται για την λειτουργία του Ringserver [25]. Παρακάτω παρατίθεται το Dockerfile για το SeedLink.

```
FROM ubuntu:20.04

RUN apt-get update -y && apt install -y gcc make

WORKDIR /seedlink
COPY. /seedlink

RUN cd /seedlink && CFLAGS="-O2" make
```

```
RUN groupadd sysop
RUN useradd -rm -d /home/sysop -s /bin/bash -g sysop -G sudo -u 1001 sysop
-p sysop

RUN passwd -d sysop

USER sysop
WORKDIR /home/sysop

EXPOSE 18000

WORKDIR /seedlink

COPY ./protocols /etc/protocols

ENTRYPOINT [ "./ringserver" ]

CMD [ "./ring.conf"]
```

Το παραπάνω Dockerfile ορίζει τη δημιουργία ενός Docker image βασισμένου στο Ubuntu 20.04. Κατά τη δημιουργία του, ενημερώνονται τα πακέτα συστήματος και εγκαθίστανται οι compilers gcc και make. Στη συνέχεια, αντιγράφονται τα αρχεία από τον τρέχοντα κατάλογο (τα αρχεία του ringserver) στον κατάλογο /seedlink του container και γίνεται compile του κώδικα. Δημιουργούνται ομάδα και χρήστης με το όνομα sysop, αφαιρείται ο κωδικός πρόσβασης για τον χρήστη sysop, και ορίζεται ο χρήστης sysop ως ο χρήστης προεπιλογής για τις επόμενες εντολές. Ορίζεται ως working directory το /home/sysop και ανοίγει η θύρα 18000 για εξωτερικές συνδέσεις. Τέλος, ορίζεται το ./ringserver ως το entry point για το container και το ./ring.conf ως η προεπιλεγμένη εντολή που θα εκτελείται όταν ξεκινά το container, αν δεν παρέχονται άλλες εντολές κατά την εκκίνηση του.

Το binary αρχείο ringserver (/seedlink-server/seedlink/ringserver) αναφέρεται στον εκτελέσιμο κώδικα του ringserver που είναι υπεύθυνο για τη διαχείριση του SeedLink ring buffer. Το αρχείο ring.conf (/seedlink-server/seedlink/ring.conf), όπως φαίνεται και παρακάτω, παρέχει τις ρυθμίσεις για τον SeedLink ring server, προκειμένου να γίνει σωστά η διαχείριση και μετάδοση των σεισμικών δεδομένων σε πραγματικό χρόνο.

```
RingDirectory /seedlink/ring

DataLinkPort 16000

SeedLinkPort 18000
```

```
ServerID "XX Seismic Network"

TransferLogDirectory /seedlink/tlog

TransferLogRX 0

MSeedScan /archive/ StateFile=/seedlink/scan.state InitCurrentState=y
```

Οι παραπάνω ρυθμίσεις επιτυγχάνουν τα εξής:

1. **Ring Directory**- Καθορίζει τον κατάλογο όπου το SeedLink θα αποθηκεύει τα δεδομένα του ring buffer, έναν προσωρινό χώρο αποθήκευσης όπου τα σεισμικά δεδομένα γράφονται και αντικαθίστανται συνεχώς με κυκλικό τρόπο.
2. **DataLinkPort**- Καθορίζει τη θύρα όπου το SeedLink θα ακούει για εισερχόμενες συνδέσεις δεδομένων από πελάτες που ζητούν σεισμικά δεδομένα.
3. **SeedLinkPort**- Καθορίζει τη θύρα όπου το SeedLink θα ακούει για εισερχόμενες συνδέσεις με το πρωτόκολλο SeedLink.
4. **ServerID**- Ορίζει το αναγνωριστικό ή το όνομα του διακομιστή SeedLink.
5. **TransferLogDirectory**- Καθορίζει τον κατάλογο όπου το SeedLink θα αποθηκεύει τα αρχεία καταγραφής για τις μεταφορές δεδομένων και τις συνδέσεις.
6. **TransferLogRX**- Ορίζει το επίπεδο λεπτομέρειας για την καταγραφή των δεδομένων που λαμβάνονται. Με την τιμή 0, υπάρχει ελάχιστη ή καθόλου καταγραφή.
7. **MSeedScan**- Ρυθμίζει το SeedLink για να αναζητά τα αρχεία δεδομένων MiniSEED στον φάκελο "/archive/" (φάκελος archive στην **Εικόνα 30**). Σε αυτόν τον φάκελο επομένως “κοιτάει” ο server SeedLink, όταν κάποιος client συνδεθεί προκειμένου να αποκτήσει πρόσβαση στα σεισμικά δεδομένα. Καθορίζει επίσης ένα αρχείο κατάστασης /seedlink/scan.state για τη διατήρηση της τρέχουσας κατάστασης της διαδικασίας scanning των MiniSEED αρχείων.

7.4.2 Slarchive

Το Slarchive ενσωματώνεται στο λογισμικό SeisComP ως ένα από τα διάφορα στοιχεία που συνθέτουν τη δομή του. Για τον λόγο αυτό, στο παρακάτω dockerfile, πραγματοποιείται η δημιουργία του συστήματος SeisComp, προκειμένου να αξιοποιηθεί η λειτουργικότητα του Slarchive. Επισημαίνεται ωστόσο ότι στόχος της παρούσας διπλωματικής δεν αποτελεί η μελέτη του εργαλείου SeisComP αλλά

μόνο η λειτουργία Slarchive που παρέχει, συνεπώς δεν γίνεται αναλυτική αναφορά σε αυτό (για περισσότερες βλ. [29]).

```
FROM ubuntu:20.04

RUN apt-get update -y && apt-get install -y sudo wget python3
libpython3.8 lsb-release nano

RUN groupadd sysop
RUN useradd -rm -d /home/sysop -s /bin/bash -g sysop -G sudo -u 1001
sysop -p sysop
RUN passwd -d sysop

USER sysop
WORKDIR /home/sysop

RUN wget 'https://www.SeisComP.de/downloader/SeisComP-4.6.0-ubuntu20.04-
x86_64.tar.gz'

RUN wget 'https://www.SeisComP.de/downloader/SeisComP-maps.tar.gz'
RUN wget 'https://www.SeisComP.de/downloader/SeisComP-4.6.0-doc.tar.gz'

RUN tar -xf SeisComP-4.6.0-ubuntu20.04-x86_64.tar.gz && rm SeisComP-
4.6.0-ubuntu20.04-x86_64.tar.gz

RUN tar -xf SeisComP-maps.tar.gz && rm SeisComP-maps.tar.gz

RUN tar -xf SeisComP-4.6.0-doc.tar.gz && rm SeisComP-4.6.0-doc.tar.gz

RUN sudo DEBIAN_FRONTEND="noninteractive" apt-get install tzdata -y

RUN sudo apt update -y


RUN sh -c /bin/echo -e "y" | sudo ./SeisComP/bin/SeisComP install-deps
base

RUN sh -c /bin/echo -e "y" | sudo ./SeisComP/bin/SeisComP install-deps
fdsnws

RUN sh -c /bin/echo -e "y" | sudo ./SeisComP/bin/SeisComP install-deps
mariadb-server

RUN mkdir /home/sysop/.SeisComP
```

```
RUN sudo chown sysop:sysop -R /home/sysop/SeisComP /home/sysop/.SeisComP

RUN sudo chmod -R 777 /home/sysop/SeisComP /home/sysop/.SeisComP

RUN sed -i 's/localhost/gisola-db/g'
/home/sysop/SeisComP/etc/defaults/scmaster.cfg

COPY ./wait-for-it.sh /home/sysop
RUN sudo chmod -R 777 ./wait-for-it.sh

RUN sudo apt-get install python3-pyqt5 -y

RUN echo "\nexport SeisComP_ROOT=\"/home/sysop/SeisComP\"\nexport
PATH=\"/home/sysop/SeisComP/bin:$PATH\"\nexport
LD_LIBRARY_PATH=\"/home/sysop/SeisComP/lib:$LD_LIBRARY_PATH\"\nexport
PYTHONPATH=\"/home/sysop/SeisComP/lib/python:$PYTHONPATH\"\nexport
MANPATH=\"/home/sysop/SeisComP/share/man:$MANPATH\"\nexport
LC_ALL=C\nsource \"/home/sysop/SeisComP/share/shell-
completion/SeisComP.bash\"\n" >> /home/sysop/.bashrc

RUN sudo apt install -y openssh-server xauth

RUN sudo mkdir /var/run/sshd \
 && mkdir /home/sysop/.ssh \
 && chmod 700 /home/sysop/.ssh \
 && sudo ssh-keygen -t rsa -N '' -f /home/sysop/.ssh/id_rsa

RUN sudo sed -i "s/^.*PasswordAuthentication.*$/PasswordAuthentication
yes/" /etc/ssh/sshd_config

RUN sudo sed -i 's/#PermitRootLogin prohibit-password/PermitRootLogin
yes/' /etc/ssh/sshd_config

RUN sudo sed -i "s/^.*X11Forwarding.*$/X11Forwarding yes/"
/etc/ssh/sshd_config

RUN sudo sed -i "s/^.*X11UseLocalhost.*$/X11UseLocalhost no/"
/etc/ssh/sshd_config

RUN sudo grep "^X11UseLocalhost" /etc/ssh/sshd_config || echo
"X11UseLocalhost no" >> /etc/ssh/sshd_config

RUN sudo apt install net-tools tightvncserver novnc -y
```

```
ENV DEBIAN_FRONTEND noninteractive

RUN sudo apt update
RUN sudo DEBIAN_FRONTEND=noninteractive apt-get install -y keyboard-
configuration
RUN sudo DEBIAN_FRONTEND=noninteractive apt-get install -y lightdm

RUN sudo apt install -y xfce4 xfce4-goodies

RUN export USER=sysop

RUN mkdir $HOME/.vnc && echo "example\nexample\n" | vncpasswd >
$HOME/.vnc/passwd && chmod 400 $HOME/.vnc/passwd

RUN echo "#!/bin/bash\nxrdb $HOME/.Xresources\nstartxfce4 &\nxfce4-
terminal" > $HOME/.vnc/xstartup && chmod +x $HOME/.vnc/xstartup

ENV USER sysop

RUN echo 'alias scolv-gisola="scolv -d mysql://sysop:sysop@gisola-
db/SeisComP"' >> ~/.bashrc

RUN wget http://old.kali.org/kali/pool/main/x/xfwm4/xfwm4_4.14.2-
1_amd64.deb && sudo dpkg -i *.deb && rm *.deb

RUN sudo apt-get clean && sudo rm -rf /var/lib/apt/lists/* /tmp/*
/var/tmp/*
```

Το παραπάνω Dockerfile δημιουργεί ένα εικονικό περιβάλλον βασισμένο στην εικόνα του Ubuntu 20.04, σχεδιασμένο για να υποστηρίξει την εγκατάσταση και ρύθμιση του λογισμικού SeisComP. Η διαδικασία ξεκινά με την ενημέρωση και την εγκατάσταση απαραίτητων πακέτων όπως sudo, wget, python3, libpython3.8 και άλλα. Στη συνέχεια, δημιουργείται ένα νέο group (sysop) και ταυτόχρονα ένας νέος χρήστης (sysop). Ο νέος χρήστης προστίθεται στο group sysop, έχει το φάκελο home του ορισμένο στο /home/sysop, χρησιμοποιεί το shell /bin/bash και του παρέχεται επίσης η δυνατότητα sudo για να έχει δικαιώματα διαχειριστή. Ο UID (αριθμός χρήστη) ορίζεται σε 1001. Τέλος, η εντολή passwd -d sysop αφαιρεί τον κωδικό πρόσβασης για τον χρήστη sysop, επιτρέποντας τη σύνδεση χωρίς κωδικό πρόσβασης.

Έπειτα πραγματοποιείται το κατέβασμα και η αποσυμπίεση κάποιων απαραίτητων πακέτων του SeisComP. Το configuration συνεχίζεται με ρυθμίσεις όσον αφορά στο SSH, στη δημιουργία του αρχείου αναμονής wait-for-it.sh, και την εγκατάσταση του περιβάλλοντος εργασίας XFCE. Επίσης,

γίνεται η προετοιμασία του συστήματος για τη χρήση του VNC server, καθιστώντας δυνατή την απομακρυσμένη επικοινωνία με το περιβάλλον του SeisComP.

Τέλος, το Dockerfile ολοκληρώνεται με τη δημιουργία κλειδιών SSH, και την εγκατάσταση απαραίτητων πακέτων για την χρήση του VNC server. Επίσης, περιλαμβάνει ρυθμίσεις για το XFCE και τη δημιουργία ενός αρχείου εκκίνησης (xstartup) για το VNC server. Καθαρίζονται προσωρινά τα αρχεία και δημιουργείται ένα εικονικό περιβάλλον έτοιμο για τη χρήση του SeisComP σε ένα περιβάλλον XFCE, με δυνατότητα προσπέλασης μέσω SSH και VNC.

Για την καλύτερη κατανόηση της δημιουργίας του SeisComP container, παρακάτω εξηγούνται κάποιες βασικές έννοιες που αναφέρθηκαν.

XFCE

XFCE [26] είναι ένα ελαφρύ, γρήγορο και ανοιχτού κώδικα περιβάλλον εργασίας για γραφικά περιβάλλοντα Unix. Είναι μια επιλογή για χρήστες που θέλουν ένα απλό, αποδοτικό, και ελαφρύ περιβάλλον εργασίας χωρίς τη βαρύτητα των πιο πλούσιων σε χαρακτηριστικά περιβαλλόντων όπως το GNOME ή το KDE.

Το XFCE προσφέρει ένα απλό και ευέλικτο περιβάλλον εργασίας με ένα μικρό αποτύπωμα μνήμης και χρήσης επεξεργαστή. Παρέχει τα βασικά στοιχεία ενός γραφικού περιβάλλοντος, όπως πίνακα εργασίας, μενού εφαρμογών, μπάρα εργαλείων, και είναι εξαιρετικά παραμετροποιήσιμο, επιτρέποντας στους χρήστες να προσαρμόσουν το περιβάλλον εργασίας στις ανάγκες τους.

Λόγω του χαμηλού απαιτούμενου υλικού και της ευελιξίας του, το XFCE είναι δημοφιλές σε συστήματα που λειτουργούν σε περιορισμένους πόρους, όπως παλαιότερους υπολογιστές ή συσκευές με περιορισμένους πόρους.

VNC

Το VNC (Virtual Network Computing) [27] είναι ένα πρωτόκολλο που επιτρέπει στους χρήστες να αποκτούν απομακρυσμένη πρόσβαση και έλεγχο σε γραφικά περιβάλλοντα εργασίας. Ο όρος VNC Server αναφέρεται στο λογισμικό που εκτελείται στον υπολογιστή που ελέγχεται απομακρυσμένα.

Ο VNC Server είναι υπεύθυνος για την δημιουργία μιας εικονικής επιφάνειας εργασίας και την παροχή της στους απομακρυσμένους χρήστες. Όταν ένας απομακρυσμένος χρήστης συνδέεται μέσω ενός VNC Client, μπορεί να δει την επιφάνεια εργασίας του συστήματος, να εκτελεί εφαρμογές και να εκτελεί ενέργειες, όλα αυτά απομακρυσμένα.

Το VNC είναι χρήσιμο για απομακρυσμένη υποστήριξη, εργασίες και γενικά για οποιαδήποτε περίπτωση όπου απαιτείται η δυνατότητα πρόσβασης και ελέγχου σε ένα σύστημα από μακρινή τοποθεσία.

Η λειτουργικότητα που προσφέρει το Slarchive, επιτυγχάνεται με τη δημιουργία των παρακάτω αρχείων για το σωστό configuration.

Καταρχάς, το αρχείο slarchive.streams (τοπικά εντοπίζεται στη διαδρομή /seismograph/slarchive/var/lib/slarchive/slarchive.streams), ο κώδικας του οποίου φαίνεται παρακάτω, αποτελεί ένα αρχείο λίστας ροών (stream list file). Χρησιμοποιείται για τον καθορισμό των ροών δεδομένων που πρέπει να ανακτηθούν από σεισμικούς σταθμούς. Κάθε γραμμή του αρχείου καθορίζει μια ροή δεδομένων από έναν σεισμικό σταθμό. Εντός του container έχει δημιουργηθεί στο path /home/sysop/slarchive/seiscomp/var/lib/slarchive. Παραδείγματος χάριν, το HP UPR HH*.D σημαίνει το εξής:

1. **HP**- Network
2. **UPR**- Station
3. **HH**- Channel
4. *****- χρησιμοποιείται ως wildcard για να αντιστοιχίσει οποιοδήποτε όνομα ροής που ακολουθεί το πρόθεμα HH
5. **.D**- Συμβολίζει ότι πρόκειται για σεισμικά δεδομένα (data)

```
---- Begin example file -----
# Comment lines begin with a '#' or '*'
# Example stream list file for use with the -l argument of slclient or
# with the sl_read_streamlist() libslink function.
# specifying the data acquired from a specific seismic station
HP UPR HH*.D

---- End example file -----
```

Συνοψίζοντας, το παραπάνω αρχείο καθορίζει, κατά την εκκίνηση του Slarchive container, την προέλευση των σεισμικών δεδομένων που θα αποθηκευτούν στο φάκελο archive (κάθε πακέτο σεισμικών δεδομένων αποθηκεύεται ως MiniSEED) και στη βάση gisola-db (βλ. Εικόνα 30).

Σημαντικό επίσης αρχείο που συνέβαλε στη σωστή ρύθμιση του Slarchive, αποτελεί το slarchive.cfg (τοπικά εντοπίζεται στη διαδρομή /seismograph/slarchive/SeisComP3/slarchive.cfg), όπως αυτό φαίνεται παρακάτω. Πιο συγκεκριμένα, αυτό είναι ένα αρχείο ρύθμισης που χρησιμοποιείται για να καθορίσει πώς το Slarchive θα πρέπει να συμπεριφέρεται, ποια δεδομένα θα πρέπει να συλλέγει, πού θα πρέπει να αποθηκεύει αυτά τα δεδομένα και άλλες ρυθμίσεις που σχετίζονται με τη λειτουργία του. Εντός του container έχει δημιουργηθεί στο path /home/sysop/.seiscomp/slarchive.cfg

```
address = 150.140.182.7
```

```
port = 18000

archive = var/lib/archive

buffer = 10

delay = 30

networkTimeout = 900

idleTimeout = 30

keepalive = 0

validation.certs = var/lib/certs

validation.mode = ignore
```

Παρακάτω εξετάζονται κάποιες από τις σημαντικές ρυθμίσεις του αρχείου:

1. **address = 150.140.182.7**: Αυτή η ρύθμιση καθορίζει τη διεύθυνση IP του Seedlink server που το Slarchive θα συνδεθεί, προκειμένου να φορτώσει τα σεισμικά δεδομένα.
2. **port = 18000**: Καθορίζει τη θύρα του Seedlink server που το Slarchive θα χρησιμοποιήσει για τη σύνδεση.
3. **archive = var/lib/archive**: Το path στον container όπου θα αποθηκευτούν τα σεισμικά δεδομένα. Σημειώνεται ότι το path που προκύπτει εν τέλει για την αποθήκευση των δεδομένων στο φάκελο archive ακολουθεί τη διάταξη SDS (βλ. 6.2.1)
4. **buffer = 10**: Ο αριθμός των εγγραφών (σε μονάδες των 512 bytes) που θα αποθηκεύονται προσωρινά πριν από την εγγραφή τους στο δίσκο.
5. **delay = 30**: Ο χρόνος (σε δευτερόλεπτα) πριν προσπαθήσει να επανασυνδεθεί στο SeedLink server σε περίπτωση διακοπής της σύνδεσης.

7.4.3 Διαδικασία Κρυπτογράφησης/Αποκρυπτογράφησης

Κατά την εκκίνηση του Slarchive container, ξεκινά να τρέχει και ένα python script ώστε να παρακολουθεί αλλαγές στον φάκελο archive όπου και αποθηκεύονται τα σεισμικά δεδομένα. Όταν εντοπιστεί μια αλλαγή (τροποποίηση) σε ένα αρχείο μέσα στον archive φάκελο, εκτελείται ο αντίστοιχος κώδικας που αφορά στην κρυπτογράφηση των δεδομένων του συγκεκριμένου αρχείου και

έπειτα ανοίγει μια σύνδεση SSL/TLS, ώστε τα κρυπτογραφημένα δεδομένα να αποσταλούν από τον client στον server. Αξίζει ωστόσο να σημειωθούν τα παρακάτω.

1. Τα σεισμικά δεδομένα που φορτώνονται μέσω του Slarchive, αποθηκεύονται στο φάκελο archive με τη μορφή MiniSEED. Κατόπιν, με τη χρήση της βιβλιοθήκης ObsPy. αυτά τα δεδομένα διαβάζονται ως ένα αντικείμενο stream. Ωστόσο, επειδή αναφερόμαστε σε ένα MiniSEED, το stream διαθέτει ένα αντικείμενο τύπου Trace (stream[0]), το οποίο και επιλέγεται για την κρυπτογράφηση των δεδομένων.
2. Το script συνεχίζει να εκτελείται συνεχώς, παρακολουθώντας τον φάκελο archive, μέχρι να διακοπεί.

7.4.3.1 Κρυπτογράφηση Σεισμικών Δεδομένων

Για την κρυπτογράφηση κάθε πακέτου σεισμικών δεδομένων δημιουργείται ξεχωριστό master key και iv, σύμφωνα με τις παρακάτω συναρτήσεις.

```
 def generate_unique_nonce(self, size):
 timestamp_bytes = int(time()).to_bytes(8, byteorder='big')
 random_bytes = token_bytes(size- 8)

 return timestamp_bytes + random_bytes

 def derive_key_and_iv(self, password, nonce):
 password_bytes = password.encode('utf-8')

 derived_key = bcrypt.kdf(password_bytes, salt=nonce,
desired_key_bytes=32, rounds=16)

 master_key = int.from_bytes(derived_key[:16], byteorder='big')
 iv = int.from_bytes(derived_key[16:28], byteorder='big')
```

Αυτές οι δύο συναρτήσεις επιτυγχάνουν τα εξής:

generate_unique_nonce(self, size):

- Αυτή η συνάρτηση δημιουργεί ένα nonce (Number used once), που είναι ένας αριθμός που χρησιμοποιείται μόνο μια φορά.
- Το nonce συνήθως χρησιμοποιείται για να εξασφαλίσει τη μοναδικότητα κάθε φορά που γίνεται κρυπτογράφηση.
- Στη συγκεκριμένη υλοποίηση, το nonce περιλαμβάνει 8 bytes του timestamp (χρόνος σε δευτερόλεπτα) και τα υπόλοιπα bytes είναι τυχαία δεδομένα που παράγονται από τη

συνάρτηση token_bytes της βιβλιοθήκης secrets. Με το πρώτο μέρος του nonce να είναι ένα timestamp μηδενίζεται η πιθανότητα να προκύψει σε κάποια μελλοντική κρυπτογράφηση των δεδομένων το ίδιο nonce.

derive_key_and_iv(self, password, nonce):

- Αυτή η συνάρτηση παίρνει έναν κωδικό πρόσβασης (password), ο οποίος φορτώνεται από ένα .env αρχείο και ένα nonce και παράγει ένα ζεύγος (key, IV) που χρησιμοποιείται στη συνέχεια για την κρυπτογράφηση.
- Χρησιμοποιεί το bcrypt για να παράγει ένα παράγει ένα κλειδί (derived key) από τον κωδικό πρόσβασης και το nonce.
- Το κλειδί αυτό (derived key) είναι μια σειρά από bytes, από τα οποία ορίζονται το master key (πρώτα 16 bytes) και ο initialization vector (IV, επόμενα 12 bytes).

Όλη αυτή η διαδικασία της δημιουργίας διαφορετικού master key και initialization vector (IV) κάθε φορά για την κρυπτογράφηση των δεδομένων συνεισφέρει σημαντικά στην ασφάλεια του συστήματος κρυπτογράφησης. Πιο συγκεκριμένα:

1. **Μοναδικότητα κλειδιού (Master Key)**- Η χρήση διαφορετικού master key κάθε φορά παρέχει μοναδικότητα στον καθορισμό του κύριου κλειδιού κρυπτογράφησης. Αυτό σημαίνει ότι, ακόμη και αν κάποιος καταφέρει να αποκτήσει πρόσβαση σε ένα παλιό master key, δεν μπορεί να το χρησιμοποιήσει για αποκρυπτογράφηση νέων δεδομένων.
2. **Αποφυγή παγίδευσης (Initialization Vector- IV)**- Το IV είναι κρίσιμο για την αποφυγή παγίδευσης σε κρυπτογραφικές λειτουργίες. Αν χρησιμοποιηθεί το ίδιο IV για διάφορες κρυπτογραφήσεις, μπορεί να υπάρξουν προβλήματα ασφαλείας. Η χρήση διαφορετικών IV καθιστά αδύνατο τον εντοπισμό προτύπων και προσθέτει ασφάλεια στην κρυπτογράφηση.

Αφότου δημιουργηθούν το master key και το iv, τα metadata του Trace μετατρέπονται στη δομή δεδομένων dictionary της python περιλαμβάνοντας τα χαρακτηριστικά του αντικειμένου Trace.stats, όπως το δίκτυο (network), ο σταθμός (station), η τοποθεσία (location) κ.ά. Έπειτα, το dictionary μετατρέπεται σε μια JSON συμβολοσειρά και κωδικοποιείται σε bytes. Αντίστοιχα, το Trace.data (τα σεισμικά δεδομένα δηλαδή) μετατρέπονται επίσης σε μορφή bytes.

Έπειτα, αφού τα metadata και data είναι σε μορφή bytes συγκεντρώνονται σε ένα κοινό byte array. Ωστόσο, χρησιμοποιείται ένας διαχωριστικός χαρακτήρας (delimiter), όπως φαίνεται στην παρακάτω γραμμή κώδικα, για να διαχωρίσει τα stats και data του Trace, με στόχο να αποφευχθεί η ξεχωριστή κρυπτογράφηση των παραπάνω δύο τμημάτων. Κατά αυτό τον τρόπο, μειώνεται ο όγκος των δεδομένων που πρέπει να κρυπτογραφηθούν και αυξάνεται η αποδοτικότητα της διαδικασίας κρυπτογράφησης.

```
trace_combined = trace_stats + self.delimiter + trace_data
```

Το delimiter (υπό μορφή bytes) χρησιμοποιείται, ώστε κατά τη διαδικασία της αποκρυπτογράφησης να απομονωθούν εύκολα τα δύο τμήματα του κρυπτογραφημένου μηνύματος, δηλαδή τα trace.stats και trace.data. Είναι σημαντικό λοιπόν να επιλεχθεί ένας διαχωριστικός χαρακτήρας για τον οποίο υπάρχει μηδενική πιθανότητα να εμφανιστεί στα δεδομένα προς κρυπτογράφηση, όπως φαίνεται και στη παρακάτω γραμμή κώδικα.

```
self.delimiter = b'##!@!##'
```

Τέλος, το κοινό byte array κρυπτογραφείται χρησιμοποιώντας τον αλγόριθμο AES-GCM. Το αποτέλεσμα της κρυπτογράφησης περιλαμβάνει τα κρυπτογραφημένα δεδομένα και το authentication tag. Το τελευταίο μετατρέπεται σε byte representation των 16 bytes και προστίθεται στο τέλος του κρυπτογραφημένου μηνύματος, όπως φαίνεται και παρακάτω.

```
auth_tag_bytes = auth_tag.to_bytes(16, byteorder='big')

encrypted_trace_with_tag = encrypted_trace + auth_tag_bytes
```

7.4.3.2 Αποκρυπτογράφηση Σεισμικών Δεδομένων

Μόλις τα δεδομένα φτάσουν στον κεντρικό Seedlink server, ξεκινά η διαδικασία της αποκρυπτογράφησης. Η παραπάνω διαδικασία επιτυγχάνεται με τη χρήση του κύριου κλειδιού (master key) και του διανύσματος αρχικοποίησης (IV), που χρησιμοποιήθηκαν κατά τη διαδικασία κρυπτογράφησης των αρχικών δεδομένων.

Αρχικά, απομονώνονται τα τελευταία 16 bytes από τα κρυπτογραφημένα δεδομένα που αντιστοιχούν στο authentication tag και μετατρέπονται σε ακέραιο αριθμό.

```
auth_tag_bytes = encrypted_trace[-16:]

auth_tag = int.from_bytes(auth_tag_bytes, byteorder='big')
```

Ακολούθως, αποκρυπτογραφούνται τα δεδομένα εξαιρουμένου του authentication tag. Ωστόσο, το τελευταίο χρησιμοποιείται στην διαδικασία της αποκρυπτογράφηση, ώστε να επιβεβαιωθεί ότι τα δεδομένα δεν έχουν αλλοιωθεί κατά τη μεταφορά τους. Αν προκύψει κάποιο σφάλμα όσον αφορά στον έλεγχο της ακεραιότητας των δεδομένων, τότε αυτά απορρίπτονται από τον server, ο οποίος προχωρά στην αίτηση του επόμενου πακέτου σεισμικών δεδομένων.

Έπειτα τα δεδομένα διαχωρίζονται με τη χρήση του delimiter, όπως φαίνεται και στην παρακάτω γραμμή κώδικα, προκειμένου να ανακτηθούν ξεχωριστά τα trace.stats και trace.data.

```
parts = combined_data.split(self.delimiter)
```

Στη συνέχεια, το trace.stats από bytes μετατρέπεται σε JSON string και έπειτα σε dictionary. Αντίστοιχα, το trace.data μετατρέπεται από μορφή bytes σε numpy array, όπως φαίνεται ενδεικτικά παρακάτω.

```
data = np.frombuffer(trace_data_binary, dtype=dtype)
```

Κατά αυτό τον τρόπο, ανακατασκευάζεται πλήρως το trace σε αυτό που στάλθηκε αρχικά και αποθηκεύεται στον αντίστοιχο φάκελο archive, σύμφωνα με το πρότυπο SDS.

7.4.4 SSL/TLS Server/Client

Ο Client και Server επικοινωνούν μεταξύ τους μέσω SSL/TLS (Secure Sockets Layer/Transport Layer Security), ώστε να διασφαλιστεί η ασφαλής και κρυπτογραφημένη επικοινωνία τους. Αυτή η επικοινωνία επιτυγχάνεται με τη χρήση του πακέτου OpenSSL και της βιβλιοθήκης PyOpenSSL.

Συγκεκριμένα. ο server δημιουργεί ένα SSL server socket και εγκαθιστά μια ασφαλή σύνδεση SSL με τον client, όταν αυτός συνδέεται. Η διαδικασία αυτή όπως θα αναλυθεί εκτενέστερα παρακάτω περιλαμβάνει τη χρήση κρυπτογραφημένων κλειδιών, TLS handshake και ανταλλαγή κρυπτογραφημένων δεδομένων.

Από την μεριά του client, εγκαθίσταται η σύνδεση SSL με τον server, ενώ ολόκληρη η ανταλλαγή δεδομένων μεταξύ client και server γίνεται μέσω αυτής της ασφαλούς σύνδεσης.

Για τη δημιουργία και χρήση διαπιστευτηρίων SSL/TLS στον server χρησιμοποιούνται δύο αρχεία .pem (Privacy Enhanced Mail) που ονομάζονται server-key-encrypted.pem και server-cert.pem. Αυτά τα αρχεία περιέχουν το ιδιωτικό κλειδί (private key) και το πιστοποιητικό (certificate) αντίστοιχα. Βρίσκονται στην κατηγορία των X.509 certificates και διαδραματίζουν καίριο ρόλο στο πλαίσιο της κρυπτογράφησης και της αυθεντικοποίησης όσον αφορά στην ασφαλή επικοινωνία μεταξύ πελάτη και εξυπηρετητή. Παρακάτω αναφέρεται αναλυτικά η διαδικασία που ακολουθήθηκε για τη δημιουργία και χρήση αυτών των αρχείων.

Αρχικά, δημιουργείται ένα ζεύγος κλειδιών (key pair), που αποτελείται από το ιδιωτικό κλειδί (server-key.pem) και το δημόσιο κλειδί (server-cert.pem). Αυτό έγινε με την παρακάτω εντολή OpenSSL:

```
openssl req -x509 -nodes -newkey rsa:2048 -keyout server-key.pem -out
server-cert.pem -days 365
```

Σε αυτό το σημείο, έχει δημιουργηθεί ένα αυτο-υπογεγραμμένο πιστοποιητικό (self-signed certificate) που προορίζεται για χρήση από τον server

Στη συνέχεια, το ιδιωτικό κλειδί κρυπτογραφείται με κωδικό πρόσβασης για περισσότερη ασφάλεια, με αποτέλεσμα να προκύψει το καινούργιο αρχείο server-key-encrypted.pem. Αυτό γίνεται με την παρακάτω εντολή OpenSSL:

```
openssl rsa -aes256 -in server-key.pem -out server-key-encrypted.pem
```

Και τα δύο αυτά αρχεία (server-key-encrypted.pem και server-cert.pem) φορτώνονται στο SSL context του server ως μέρος της διαμόρφωσης του περιβάλλοντος SSL.

Κατά τη διάρκεια του TLS handshake, ο server στέλνει το δημόσιο κλειδί του στον client.. Μετά την αποστολή λοιπόν του δημόσιου κλειδιού, ο client δημιουργεί ένα αντικείμενο κρυπτογράφησης (cipher) και με τη χρήση του αλγορίθμου RSA κρυπτογραφεί τις αρχικές απαραίτητες πληροφορίες που θα αποστείλει στον server και χρειάζονται πριν αποσταλούν τα κρυπτογραφημένα (μέσω AES-GCM) σεισμικά δεδομένα. Έπειτα, οι παραπάνω πληροφορίες στέλνονται στον server με τη χρήση του ασφαλούς SSL socket, ενώ ο server τις λαμβάνει και αποκρυπτογραφεί με τη βοήθεια του ιδιωτικού κλειδιού.

Επισημαίνεται ότι για κάθε πακέτο σεισμικών δεδομένων που δημιουργείται από τον "σεισμογράφο", ανοίγει μια σύνδεση SSL/TLS και όταν ολοκληρωθεί η επικοινωνία server- client, αυτή η σύνδεση τερματίζεται. Η παραπάνω επιλογή έγινε για τους εξής λόγους:

1. **Ασφάλεια Δεδομένων:** Κλείνοντας τη σύνδεση μετά την επιτυχή μεταφορά δεδομένων, μειώνεται ο χρόνος κατά τον οποίον ένας επιτιθέμενος θα μπορούσε να προσπελάσει τη σύνδεση και να παρακολουθήσει την επικοινωνία.
2. **Προστασία από Επιτιθέμενους:** Αν η σύνδεση παραμείνει ανοιχτή, υπάρχει περισσότερος χρόνος για τους επιτιθέμενους να προσπαθήσουν να αποκτήσουν πρόσβαση στα δεδομένα ή να πραγματοποιήσουν επιθέσεις όπως η επίθεση man-in-the-middle.
3. **Εξοικονόμηση Πόρων:** Η διακοπή της σύνδεσης μειώνει τη χρήση ενεργειακών και υπολογιστικών πόρων, ιδιαίτερα σε περιβάλλοντα με περιορισμένους πόρους.

Τέλος, από την πλευρά του server έχει δημιουργηθεί το αρχείο allowed_ips.conf, το οποίο έχει ως στόχο τον περιορισμό των IP που επιτρέπεται να συνδεθούν στον server. Πιο συγκεκριμένα,

1. Το αρχείο αυτό πρέπει να περιέχει μια διεύθυνση IP ανά γραμμή.
2. Ο server διαβάζει το αρχείο και φορτώνει τις διευθύνσεις IP σε μια λίστα.
3. Αν η λίστα είναι άδεια ή το αρχείο δεν υπάρχει, τότε η λίστα των επιτρεπόμενων διευθύνσεων είναι κενή.

Έπειτα, κατά τη διαδικασία σύνδεσης ενός client, ο server ελέγχει αν η διεύθυνση IP του client περιλαμβάνεται στη λίστα των επιτρεπόμενων διευθύνσεων. Αν η διεύθυνση είναι επιτρεπτή, ο client επιτρέπεται να συνδεθεί. Σε διαφορετική περίπτωση, η σύνδεση απορρίπτεται.

7.4.4.1 Αποστολή Δεδομένων μέσω SSL/TLS

Αφού το handshake μεταξύ client- server ολοκληρωθεί επιτυχώς, στέλνονται τα παρακάτω δεδομένα.

1. Ο client υπολογίζει το συνολικό μέγεθος των κρυπτογραφημένων δεδομένων και το στέλνει στον server, ώστε ο τελευταίος να γνωρίζει εκ των προτέρων πόσα δεδομένα πρέπει να αναμένει από τον client. Αυτό είναι απαραίτητο για να μπορέσει να εξασφαλίσει ότι θα λάβει όλα τα δεδομένα που αναμένει και ότι δεν θα υπάρξει απώλεια ή επιπλέον προσθήκη κατά τη μετάδοση τους (αποφυγή man in the middle attack, length extension attack κ.ο.κ.). Ακόμη, η γνώση του συνολικού μεγέθους των δεδομένων επιτρέπει στον server να προετοιμαστεί για την επεξεργασία των δεδομένων καθώς αυτά φτάνουν. χωρίς να χρειάζεται να περιμένει διότι δεν γνωρίζει πότε θα ολοκληρωθεί η μετάδοση.

2. Ο client υπολογίζει το τμήμα (chunk) των δεδομένων που θα στέλνει κάθε φορά επιτρέποντάς του να μοιράσει τα δεδομένα σε μικρά τμήματα. Αυτό επιτρέπει στον server να απαντά συνεχώς κατά τη διάρκεια λήψης των δεδομένων, μειώνοντας την ολική καθυστέρηση. Επιπλέον, ο server μπορεί να διαχειριστεί αποτελεσματικότερα τη μνήμη, καθώς δεν χρειάζεται να δεσμεύσει χώρο για όλα τα δεδομένα ταυτόχρονα. Άλλωστε, γνωρίζει ήδη πόσα τμήματα πρέπει να περιμένει, αφού του έχει αποσταλεί προηγουμένως το συνολικό μέγεθος των δεδομένων που πρέπει να αναμένει. Ο υπολογισμός του chunk γίνεται με βάση το συνολικό μέγεθος των δεδομένων και το μέγιστο μέγεθος του κάθε chunk, το οποίο είναι πλήρως παραμετροποιήσιμο ωστόσο στην παρούσα υλοποίηση επιλέχθηκε το 8192.

```
def calculate_chunk_size(self, total_size, max_chunk_size):
num_chunks = (total_size + max_chunk_size- 1) // max_chunk_size

chunk_size = total_size // num_chunks

return chunk_size
```

3. Ο client στέλνει τα ίδια τα κρυπτογραφημένα σεισμικά δεδομένα στον server σε chunks. Η διαδικασία αυτή επαναλαμβάνεται μέχρι να έχουν σταλεί όλα τα δεδομένα.

```
def send_chunk_encrypted_data(self, total_data_size,chunk_size):
index = 0
```

```
while index < total_data_size:
chunk = self.encrypted_data[index:(index + chunk_size)]

self.ssl_socket.send(chunk)

index += len(chunk)
```

Επισημαίνεται ότι οι πληροφορίες στα βήματα 1 και 2, πριν την αποστολή τους στον server, κρυπτογραφούνται από τον client με τη χρήση του δημόσιου κλειδιού που του είχε σταλεί προηγουμένως και τον αλγόριθμο RSA. Επιπλέον, η μετάδοση όλων των δεδομένων γίνεται μέσω του SSL socket.

8

Συμπεράσματα και Προοπτικές

8.1 Σύνοψη

Στο πλαίσιο της παρούσας διπλωματικής εργασίας, σχεδιάστηκε και αναπτύχθηκε μία νέα αρχιτεκτονική βασισμένη στο ήδη υπάρχον σύστημα απόκτησης και μεταφοράς σεισμικών δεδομένων. Η αρχιτεκτονική αυτή αναπτύχθηκε με την χρήση της γλώσσα προγραμματισμού Python και της τεχνολογίας containers (μέσω Docker) ενώ αξιοποιήθηκαν και σημαντικές τεχνολογίες στο χώρο της σεισμολογίας, όπως SeedLink, Slarchive, SeisComP και Ringserver. Κύριος στόχος ήταν η ενσωμάτωση ενός αξιόπιστου μηχανισμού κρυπτογραφίας για τα σεισμικά δεδομένα που μεταφέρονται μέσω του πρωτοκόλλου SeedLink, προσφέροντας κατά αυτόν τον τρόπο ανθεκτικότητα σε επιθέσεις όπως man-in-the-middle attack. Η προτεινόμενη αρχιτεκτονική λοιπόν διασφαλίζει στους ερευνητές ότι τα δεδομένα που λαμβάνουν, είναι ασφαλή και ακέραια χωρίς ωστόσο να επηρεάζει αισθητά τα εργαλεία τα οποία λειτουργούν μέχρι στιγμής,

8.2 Συμπεράσματα

Συνολικά, η προτεινόμενη αρχιτεκτονική της παρούσας διπλωματικής αποτελεί μια πρώτη προσπάθεια για μια καινοτόμα προσέγγιση στην επιστήμη της Σεισμολογίας και των Υπολογιστών όσον αφορά στην ασφαλή μεταφορά δεδομένων μέσω του πρωτοκόλλου SeedLink. Διαδεχόμενη την πρόταση αυτή, αποδεικνύει ότι η ενσωμάτωση μηχανισμού ασφάλειας είναι εφικτή σε ένα τέτοιο σύστημα, το οποίο μεταφέρει δεδομένα σε πραγματικό χρόνο, συνδυάζοντας την ταχεία μετάδοση με τη διατήρηση της

εμπιστευτικότητας και ακεραιότητας των πληροφοριών. Ακόμη, ανοίγει τον δρόμο για επιπλέον έρευνα και υιοθέτηση μέτρων ασφάλειας, με στόχο τη θωράκιση του πρωτοκόλλου SeedLink από πιθανές επιθέσεις οποιουδήποτε είδους και την ενίσχυση της αξιοπιστίας του. Η παρούσα υλοποίηση ακόμη επιτυγχάνει σημαντικά οφέλη όπως αυτά εξηγούνται παρακάτω.

1. **Μη απαιτούμενες αλλαγές στο hardware του Data Reception System:**

   Δεν απαιτούνται πρόσθετες αλλαγές στο hardware του Data Reception System, που μεταφέρει τα σεισμικά δεδομένα στον κεντρικό server του σεισμικού σταθμού.

2. **Απομάκρυνση του πρωτοκόλλου SeedLink:**

   Η ανάγκη για εγκατάσταση του πρωτοκόλλου SeedLink στο Data Reception System είναι πλέον περιττή. Η μεταφορά των δεδομένων πραγματοποιείται μέσω μιας ασφαλούς σύνδεσης SSL/TLS, με αποτέλεσμα τη μείωση της υπολογιστικής ισχύος που απαιτείται.

3. **Απομακρυσμένη εγκατάσταση νέων στοιχείων:**

   Η εγκατάσταση νέων συστατικών στο Data Reception System, που ήδη βρίσκεται σε λειτουργία, μπορεί να εκτελεστεί απομακρυσμένα μέσω μιας ασφαλούς σύνδεσης SSH.

8.2 Σύνδεσμος Github Υλοποίησης

https://github.com/mstephanidhs/Diploma-Thesis

8.3 Μελλοντική Εργασία

Η νέα γενιά σεισμολογικών οργάνων που λειτουργούν ως συσκευές IoT [31] επιτρέπουν την απόκτηση δεδομένων σχεδόν σε πραγματικό χρόνο. Αυτό προσφέρει σημαντικά πλεονεκτήματα στους σεισμολογικούς οργανισμούς όσον αφορά στην έγκαιρη καταγραφή και στον προσδιορισμό σημαντικών σεισμών, μετασεισμικών ακολουθιών, εδαφικής κίνησης και έντασης σεισμού, καθώς και δομικής απόκρισης κρίσιμων υποδομών. Επιπλέον, οι πληροφορίες για την πηγή, το μέγεθος και τη θέση του σεισμού προκύπτουν από την ταχύτατη επεξεργασία των πρώιμων σεισμικών κυμάτων που συλλέγονται όσο το δυνατόν πιο κοντά στην πηγή. Αυτά μπορούν να μεταδοθούν σε πιο απομακρυσμένες τοποθεσίες πριν από την άφιξη της ισχυρής δόνησης, παρέχοντας μια γρήγορη τηλεμετρία. Έτσι, στα δευτερόλεπτα που ακολουθούν την έναρξη του σεισμού, ένα σύστημα έγκαιρης προειδοποίησης σεισμού (earthquake early warning system- EEWS) μπορεί να παρέχει ένα αυτοματοποιημένο μήνυμα συναγερμού.

Τυχόν λανθασμένοι συναγερμοί EEWS θα υπονομεύσουν την εμπιστοσύνη σε κάθε ένα από αυτά τα συστήματα. Πιο συγκεκριμένα, εάν ένα σύστημα εκδίδει πάρα πολλούς ψευδείς συναγερμούς, τότε οι πολίτες θα αγνοήσουν τις μελλοντικές προειδοποιήσεις. Ψευδείς συναγερμοί μπορεί να προκύψουν ως αποτέλεσμα της έκδοσης πρώιμων συναγερμών χωρίς επαρκή κάλυψη δεδομένων. Ακόμη, εάν κάποια κακόβουλη επίθεση επηρεάσει τα ακατέργαστα σεισμολογικά δεδομένα χωρίς να διαταραχθεί η απόκτηση δεδομένων σε σχεδόν πραγματικό χρόνο, θα μπορούσε ακόμη και να αλλοιώσει τις υπολογιζόμενες σεισμολογικές παραμέτρους (π.χ. προέλευση, θέση, μέγεθος και παράμετροι εδαφικής κίνησης) χωρίς να γίνουν αρχικά αντιληπτές από την αρμόδια σεισμολογική υπηρεσία.

Επομένως, καθίσταται προφανές ότι η ενσωμάτωση ασφάλειας σε κάθε στάδιο του συστήματος μεταφοράς σεισμικών δεδομένων, από τον σεισμογράφο έως το λογισμικό επεξεργασίας αυτών των δεδομένων, αποτελεί αναγκαία προϋπόθεση. Η παρούσα διπλωματική εργασία απλά αποτελεί μια αφετηρία για περαιτέρω έρευνα, αλλά και πρόκληση για την εφαρμογή πρόσθετων μεθόδων ασφάλειας που θα ενισχύσουν το προαναφερθέν σύστημα. Ορισμένες προτάσεις αναφέρονται ενδεικτικά παρακάτω [23].

1. Συνεχής βελτίωση της ασφάλειας των IoT συσκευών που χρησιμοποιούνται στην σεισμολογία, μέσω τακτικών firmware ενημερώσεων.
2. Αλλαγή του προεπιλεγμένου κωδικού πρόσβασης σύνδεσης κατά την πρώτη χρήση της IoT συσκευής
3. Χρήση πρωτοκόλλων κρυπτογραφίας όπως HTTPs, Secure File Transfer FTP, Secure Shell (SSH), telnet μέσω SSH
4. Τμηματοποίηση Δικτύου (Network Segmentation)
5. Χρήση τειχών προστασίας (firewalls)
6. Χρήση εικονικού ιδιωτικού δικτύου (VPN)
7. Σύστημα πρόληψης και ανίχνευσης εισβολών
8. Μηχανισμός αντιμετώπισης DOS/DDOS επιθέσεων

Όσον αφορά στην προτεινόμενη αρχιτεκτονική της παρούσας διπλωματικής εργασίας, υπάρχει σίγουρα περιθώριο για περαιτέρω βελτιστοποιήσεις. Πιο συγκεκριμένα:

- Η υλοποίηση λαμβάνει υπόψιν την περίπτωση μόνο μίας σεισμολογικής συσκευής που συνδέεται στο κεντρικό Data Reception System ενός σταθμού, ωστόσο σε πραγματικές συνθήκες αυτό δεν ισχύει. Συνεπώς, μια βελτιστοποίηση θα μπορούσε να επικεντρωθεί στην αποτελεσματική διαχείριση ενός μεγάλου αριθμού συσκευών (load balancing).

- Όταν το Data Reception System διαπιστώνει τη μη αυθεντικότητα των δεδομένων που λαμβάνει από το σεισμολογικό όργανο, λόγω δυνητικής παρέμβασης στη σύνδεση και αλλοίωσης των δεδομένων, τα τελευταία απορρίπτονται από τον server χωρίς, ωστόσο, να ζητηθεί η εκ νέου αποστολή τους. Παρ' όλα αυτά, η συλλογή όλων ανεξαιρέτως των σεισμικών δεδομένων είναι ουσιαστική, προκειμένου να επιτραπεί η ανάλυση των δεδομένων για την πραγματοποίηση εμπεριστατωμένων μελετών από τους ερευνητές και τα σεισμικά συστήματα.
- Το Data Reception System μπορεί να υιοθετήσει ένα συνδυασμό μέτρων ασφάλειας για την θωράκιση του από επιθέσεις DOS/DDOS που μπορούν να τον θέσουν μη διαθέσιμο, όπως φιλτράρισμα και επιτήρηση κίνησης (traffic filtering/monitoring), υπηρεσίες CDN (Content Delivery Network), περιορισμός συνδέσεων (connection limiting) κ.α.

Βιβλιογραφία- Αναφορές

[1]. National Institute of Standards and Technology. (2001, November). *FIPS Publication 197: Advanced Encryption Standard (AES)*. U.S. Department of Commerce/National Institute of Standards and Technology. http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf

[2]. Tiessen, T., Knudsen, L. R., Kölbl, S., & Lauridsen, M. M. (Year). *Security of the AES with a Secret S-box*. DTU Compute, Technical University of Denmark, Denmark. https://eprint.iacr.org/2015/144.pdf

[3]. Stallings, W. (2006). *Cryptography and Network Security* (4th ed.) Upper Saddle River, NJ: Prentice Hall

[4]. Kessler, G. C. (1998, May). *An Overview of Cryptography.* Retrieved November 17, 2006.

[5]. Schneier, B. (1996). *Applied Cryptography, Second Edition: Protocols, Algorithms, and Source Code in C (cloth)*. John Wiley & Sons, Inc.

[6]. Benvenuto, C. J. (2012, May 31). *Galois Field in Cryptography*.

[7]. Glasby, S. P. (1999). Extended Euclid's algorithm via backward recurrence relations. Mathematics Magazine, 72(3), 228–230.

[8]. National Institute of Standards and Technology. (2007, November). *NIST Special Publication 800-38D: Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC* https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38d.pdf

[9]. McGrew, D., & Viega, J. (2005, May 31). *The Galois/Counter Mode of Operation (GCM).* http://csrc.nist.gov/publications/nistpubs/800-38a/sp800-38a.pdf

[10]. Sklavos, N., & Zhang, X. (2007, March 30). *Wireless Security and Cryptography: Specifications and Implementations*.

[11]. IRIS Data Management Center. (n.d.). *SeedLink Protocol*. Retrieved from https://ds.iris.edu/ds/nodes/dmc/services/SeedLink/

[12]. SeisComP. (n.d.). *SeedLink Protocol*. Retrieved from https://www.SeisComP.de/doc/apps/SeedLink.html

[13]. SeisComP. (n.d.). *SeedLink Archive*. Retrieved from https://www.SeisComP.de/doc/apps/slarchive.html

[14]. Docker. (n.d.). *Docker Engine Documentation*. Retrieved from https://docs.docker.com/engine/

[15]. Downey, A. B. (2002). *Think Python: How to Think Like a Computer Scientist* (2nd ed.). O'Reilly Media

[16]. Python Software Foundation. (n.d.). *Python 3 Documentation*. Retrieved from https://docs.python.org/3/

[17]. PyOpenSSL Project. (n.d.). *PyOpenSSL Documentation*. Retrieved from https://www.pyopenssl.org/en/latest/

[18]. ObsPy Development Team. (n.d.). *ObsPy Documentation*. Retrieved from https://docs.ObsPy.org/

[19]. Python Software Foundation. (n.d.). *secrets — Generate secure random numbers for managing secrets*. Retrieved from https://docs.python.org/3/library/secrets.html

[20]. Pycryptodome Project. (n.d.). *PyCryptodome Documentation: API reference*. Retrieved from https://pycryptodome.readthedocs.io/en/latest/src/api.html

[21]. Seismology Research Centre (a division of ESS Earth Sciences). (n.d.). *SeedLink Seismic Data Streams*. Retrieved from https://www.src.com.au/SeedLink-seismic-data-streams/?fbclid=IwAR1O8teNXVNS0OPtQ65JNmViTajDH4yADOJlskHEw8ycEIeFUWRq619guk8

[22]. Grassi, P. A., Garcia, M. E., & Fenton, J. L. (2017, June). *NIST Special Publication 800-63-3: Digital Identity Guidelines*. National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-63-3

[23]. Samios, M., Evangelidis, C. P., & Serrelis, E. (2021). Assessment of Information Security Vulnerabilities in Common Seismological Equipment. *Seismological Research Letters, 92*(2A), 933–940. https://doi.org/10.1785/0220200151

[24]. SeisCode (n.d.). *RingServer Manual*. Retrieved from https://seiscode.iris.washington.edu/projects/ringserver/wiki/Ringserver_manual

[25]. EarthScope. (n.d.). RingServer. Retrieved from https://github.com/EarthScope/ringserver

[26]. Xfce Documentation Project. (n.d.). *Xfce- The Missing Manual*. Retrieved from https://xfce-the-missing-manual.readthedocs.io/en/latest/

[27]. RealVNC. (n.d.). *Overview*. Retrieved from https://www.realvnc.com/en/developer/docs/latest/overview.html

[28]. SeisComP. (n.d.). *SeisComP GitHub Repository*. Retrieved from https://github.com/SeisComP

[29]. SeisComP. (n.d.). *SeisComP Documentation*. Retrieved from https://www.SeisComP.de/doc/

[30]. Python Watchdog Project. (n.d.). *Python Watchdog Documentation*. Retrieved from https://python-watchdog.readthedocs.io/en/stable/

[31]. Raman, C., & Raj, P. (2017). *The Internet of Things*. CRC Press.

[32]. Suarez, G., van Eck, T., Giardini, D., Ahern, T., Butler, R., & Tsuboi, S. (2008, September). *The International Federation of Digital Seismograph Networks (FDSN): An Integrated System of Seismological Observatories.*