Υλοποίηση και πειραματική αξιολόγηση μεθόδων κρυπτογράφησης για ασφαλή συστήματα υπολογιστικής νέφους

Η Μεταπτυχιακή Διπλωματική Εργασία

υποβάλλεται στην ορισθείσα

από την Συνέλευση

του Τμήματος Μηχανικών Η/Υ και Πληροφορικής

Εξεταστική Επιτροπή

από τον

Ανάργυρο Κατσουλιέρη

ως μέρος των υποχρεώσεων για την απόκτηση του

ΔΙΠΛΩΜΑΤΟΣ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ

ΣΤΗ ΜΗΧΑΝΙΚΗ ΔΕΔΟΜΕΝΩΝ ΚΑΙ ΥΠΟΛΟΓΙΣΤΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΜΕ ΕΙΔΙΚΕΥΣΗ

ΣΤΑ ΠΡΟΗΓΜΕΝΑ ΥΠΟΛΟΓΙΣΤΙΚΑ ΣΥΣΤΗΜΑΤΑ

Πανεπιστήμιο Ιωαννίνων

Φεβρουάριος 2021

Εξεταστική Επιτροπή:

- **Στέργιος Αναστασιάδης**, Αναπλ. Καθηγητής, Τμήμα Μηχανικών Η/Υ και Πληροφορικής, Πανεπιστήμιο Ιωαννίνων (Επιβλέπων)
- **Βασίλειος Δημακόπουλος**, Αναπλ. Καθηγητής, Τμήμα Μηχανικών Η/Υ και Πληροφορικής, Πανεπιστήμιο Ιωαννίνων
- **Λουκάς Γεωργιάδης**, Αναπλ. Καθηγητής, Τμήμα Μηχανικών Η/Υ και Πληροφορικής, Πανεπιστήμιο Ιωαννίνων

Contents

List of Figures

Abstract

Ανάργυρος Κατσουλιέρης, Δ.Μ.Σ. στη Μηχανική Δεδομένων και Υπολογιστικών Συστημάτων, Τμήμα Μηχανικών Η/Υ και Πληροφορικής, Πολυτεχνική Σχολή, Πανεπιστήμιο Ιωαννίνων, Φεβρουάριος 2021.
Υλοποίηση και πειραματική αξιολόγηση μεθόδων κρυπτογράφησης για ασφαλή συστήματα υπολογιστικής νέφους.
Επιβλέπων: Στέργιος Β. Αναστασιάδης, Αναπληρωτής Καθηγητής.

Το πρόβλημα το οποίο επιλύουμε στην παρούσα υλοποίηση είναι αυτό της ασφαλούς κρυπτογράφησης και αποθήκευσης δεδομένων. Πιο συγκεκριμένα, απαιτείται να παρέχονται εγγυήσεις ασφαλείας που αφορούν τον ίδιο τον πάροχο του μηχανήματος που υλοποιεί την κρυπτογράφηση των δεδομένων. Ένα βασικό μειονέκτημα αντίστοιχων υπαρχουσών υλοποιήσεων είναι πως βασίζονται σε λογισμικό επομένως παρέχονται χαμηλότερου επιπέδου εγγυήσεις. Αυτό συμβαίνει καθώς τα κλειδιά που χρησιμοποιούνται για την κρυπτογράφηση των δεδομένων μπορούν να γίνουν γνωστά στον διαχειριστή του συστήματος. Στην υλοποίηση την οποία παρουσιάζουμε η κρυπτογράφηση γίνεται με χρήση της τεχνολογίας Intel SGX η οποία βασίζεται εξ ολοκλήρου σε εξειδικευμένο υλικό. Επιπρόσθετα, υλοποιούμε λύση που βασίζεται σε ομομορφική κρυπτογράφηση δεδομένων με σκοπό να εξετάσουμε μια πλέον σύγχρονη και ανερχόμενη τεχνική που βασίζεται στο λογισμικό. Τέλος, αξιοποιήσαμε τον αλγόριθμο συμμετρικής κρυπτογράφησης AES ο οποίος εκμεταλλεύεται, για λόγους απόδοσης, το υπάρχον υλικό. Καταλήγουμε με ποιοτική και ποσοτική αξιολόγηση των παραπάνω υλοποιήσεων ώστε να παραχθούν χρήσιμα συμπεράσματα.

Extended Abstract

Anargyros Katsoulieris, M.Sc. in Data and Computer Systems Engineering, Department of Computer Science and Engineering, School of Engineering, University of Ioannina, Greece, February 2021.
Implementation and experimental evaluation of cryptographic methods for secure cloud computing systems.
Advisor: Stergios V. Anastasiadis, Associate Professor.

The problem that we solve in the current Master's thesis is that of secure data encryption and storage. More specifically, it is required to provide security guarantees concerning the vendor of the machine that implements the data encryption. A major disadvantage of similar existing implementations is that they are software-based and as a result lower-level guarantees are provided. This happens because the keys used to encrypt the data can be made known to the system administrator. In the implementation that we present the data encryption is done by using a new technology called Intel SGX which is based entirely on specialized hardware. In addition, we implement a solution based on fully uniform data ecryption in order to examine a state-of-the-art and up-to-date software-based technique. Finally, we used the an AES, a symmetric encryption algorithm, which utilizes, for performance reasons, the existing hardware. Finally, a qualitative and a quantitative evaluation of the above implementations is presented in order to produce useful conclusions.

Chapter 1

Εισαγωγη

Ένας σημαντικός τομέας της σύγχρονης επιστήμης των υπολογιστών είναι η ασφαλής αποθήκευση, επεξεργασία και διαχείριση δεδομένων και πληροφοριών. Συγκριτικά με παλιότερες δεκαετίες ο όγκος των δεδομένων που παράγονται καθημερινά είναι τάξεις μεγέθους μεγαλύτερος. Ενδεικτικά, παράγονται 2.5 τετράκις εκατομμύρια byte δεδομένων κάθε μέρα και μάλιστα ο καθημερινός ρυθμός αύξησης του μεγέθους των παραγόμενων δεδομένων είναι επιταχυνόμενος. Μάλιστα έως το 2025 ο συνολικός όγκος δεδομένων αναμένεται να αυξηθεί σε 163 zettabytes [?]. Τα δεδομένα αυτά ενώ παραμένουν ιδιωτικά, με την έννοια ότι ανήκουν στον ιδιοκτήτη τους, πολύ συχνά χρειάζεται να είναι κοινόχρηστα. Για παράδειγμα κοινόχρηστα έγγραφα μπορεί να είναι απαραίτητο να είναι προσβάσιμα από διάφορους εργαζόμενους σε μια εταιρεία. Η κοινοχρησία ορίζεται ως ο διαμοιρασμός, η συνιδιοκτησία και η αξιοποίηση των δεδομένων με τρίτους. Θα πρέπει ο ιδιοκτήτης ή οι ιδιοκτήτες των δεδομένων να είναι σε θέση να επιλέγουν τον βαθμό στον οποίο θα πραγματοποιούνται οι παραπάνω τρεις ενέργειες όπως και επίσης να υπάρχει η δυνατότητα ανάκλησης τους. Ο όρος ιδιωτικότητα αναφέρεται στο δικαίωμα ενός ατόμου ή μιας ομάδας, να αποφασίζουν αυτοβούλως και ανά πάσα στιγμή μέχρι ποιο σημείο οι πληροφορίες που αφορούν αυτούς θα διαβιβάζονται σε άλλους. Διακρίνονται διαφορετικά επίπεδα ιδιωτικότητας. Για παράδειγμα πληροφορίες όπως το ιατρικό ιστορικό ενός ασθενούς είναι εξαιρετικά ευαίσθητες και θα πρέπει αυστηρά να είναι γνωστές μόνο στον ασθενή και σε όποια άτομα εκείνος επιλέγει. Λιγότερο ευαίσθητες πληροφορίες όπως για παράδειγμα το αγαπημένο χρώμα ενός ατόμου είναι μεν

ιδιωτικές αλλά η προστασία τους πιθανό να μην έχει την ίδια αξία με τις πληροφορίες που σχετίζονται με την υγεία του. Φυσικά οποιαδήποτε διαρροή ιδιωτικών δεδομένων μπορεί να οδηγήσει σε δυσάρεστες συνέπειες για το άτομο όπως κοινωνικό σχολιασμό ή ακόμα και σε στιγματισμό. Ένα επίκαιρο παράδειγμα αποτελεί ο στιγματισμός ατόμων και κατά επέκταση οικογενειών ή και ακόμα ολόκληρων κοινωνικών ομάδων λόγω τη διασποράς της νόσου SARS-CoV-2 [?]. Πληροφορίες που μπορούν να οδηγήσουν σε στιγματισμό πρέπει να παραμένουν ασφαλείς και μυστικές.

Υπάρχουν πλατφόρμες στο νέφος οι οποίες απαιτείται να συνδυάζουν απόδοση, ασφάλεια και υψηλή διαθεσιμότητα. Όπως είναι κατανοητό χρειάστηκαν χρόνια για έρευνα και για την ανάπτυξη τους. Οι πιο δημοφιλείς υπηρεσίες είναι το Google Cloud, το Microsoft Azure, το Amazon Web Services (AWS) κλπ. Μέσω αυτών των υπηρεσιών ο χρήστης έχει την δυνατότητα να ανεβάσει και να αποθηκεύσει τα δεδομένα ή ακόμα και να τα μοιραστεί με άλλους χρήστες εφόσον το επιθυμεί. Όλα τα δεδομένα δεν αποθηκεύονται τοπικά στην εκάστοτε συσκευή του χρήστη αλλά στους υπολογιστές που αποτελούν το υπολογιστικό νέφος. Με αυτόν τον τρόπο δεν απαιτείται να έχει ο χρήστης διαθέσιμο μεγάλο αποθηκευτικό χώρο και επιπλέον τα δεδομένα του είναι διαθέσιμα σε όλες τις συσκευές με πρόσβαση στο διαδίκτυο, κινητές ή μη. Επομένως το φορτίο της αποθήκευσης καθώς και της ασφάλειας των δεδομένων μετατοπίζεται σε υπηρεσίες όπως αυτές που αναφέρθηκαν. Αν και η λύση του ζητήματος της αποθήκευσης ακούγεται σχετικά προφανής, καθώς απαιτεί αποθηκευτικά μέσα, αυτή της ασφάλειας δεν είναι και τόσο. Υπάρχουν διαφορετικές προσεγγίσεις από τους παρόχους για την επίλυση αλλά και διαφορετικές απαιτήσεις ανάμεσα στους χρήστες, μάλιστα αυτό το φαινόμενο συναντάται ακόμα και σε χρήστες της ίδιας υπηρεσίας.

Το ζήτημα λοιπόν είναι ο τρόπος με τον οποίο μπορεί κανείς να προστατεύσει τέτοιες πληροφορίες. Ειδικότερα, από την στιγμή που τα δεδομένα ενός ατόμου δεν αποθηκεύονται σε μια τοπική συσκευή αλλά σε κάποια άγνωστη ή άγνωστες συσκευές το πρόβλημα αυτό φαίνεται να γίνεται ακόμη πιο δυσεπίλυτο. Μια κοινή και επιστημονικά αποδεκτή πρακτική είναι η κρυπτογράφηση των δεδομένων ώστε αυτά να μην είναι προσβάσιμα από κάποια εξωτερική οντότητα. Η κρυπτογράφηση αποτελεί μια διαδικασία μετατροπής των δεδομένων από την αρχική τους μορφή σε μια νέα ακατάληπτη μορφή με σκοπό να μην είναι κατανοητή από κάποιον στον οποίο με οποιονδήποτε τρόπο γίνουν διαθέσιμα. Την νέα μορφή των δεδομένων

αφού την παραλάβει ο ιδιοκτήτης τους, και μόνο εκείνος, μπορεί να την αποκρυπτογραφήσει με κατάλληλο τρόπο ώστε να τα μετατρέψει στην αρχική τους μορφή κι να τα κατανοήσει. Τόσο η διαδικασία της κρυπτογράφησης όσο και αποκρυπτογράφησης βασίζονται πάνω σε μαθηματικά μοντέλα και αλγορίθμους. Οι πιο διαδεδομένοι τρόποι κρυπτογράφησης δεδομένων είναι είτε με συμμετρικό είτε με ασύμμετρο κλειδί.

Στην παρούσα εργασία παρουσιάζονται υλοποιήσεις που βασίζονται τόσο στο λογισμικό όσο και στο υλικό. Πιο συγκεκριμένα όσον αφορά τους αλγορίθμους λογισμικού, έχουμε υλοποιήσει λύση που εκμεταλλεύεται τις ιδιότητες της ομομορφικής κρυπτογράφησης. Επιπλέον, έχει υλοποιηθεί λύση που βασίζεται στον αλγόριθμο κρυπτογράφησης συμμετρικού κλειδιού Advanced Encryption Standard (AES). Αντίστοιχα, έχουμε υλοποιήσει λύση με χρήση της τεχνολογίας Intel SGX [?] η οποία χρησιμοποιεί ειδικά επιπρόσθετα κυκλώματα τα οποία πραγματοποιούν κρυπτογραφικές διαδικασίες. Ειδικότερα οι τεχνολογίες που χρησιμοποιούν το υλικό μπορούν να αποτελέσουν μια αποδεκτή λύση σε συστήματα υπολογιστικού νέφους όπου διάφοροι χρήστες έχουν την δυνατότητα να διαμοιράζονται δεδομένα με άλλους χρήστες της εκάστοτε πλατφόρμας.

Chapter 2

Υποβαθρο και Σχετικες Υλοποιησεις

2.1 Υπόβαθρο και Σχετικές Υλοποιήσεις

Στο παρόν κεφάλαιο θα παρουσιαστούν τεχνολογίες που χρησιμοποιήθηκαν στην παρούσα εργασία καθώς και ήδη υπάρχουσες σχετικές υπηρεσίες. Στόχος είναι να παρουσιαστεί το απαραίτητο υπόβαθρο για να είναι περισσότερο κατανοητός ο σχεδιασμός της υπηρεσίας.

2.2 Η ανάγκη αποθήκευσης δεδομένων στο διαδίκτυο

Η ανάγκη για αποθήκευση δεδομένων στο νέφος δεν είναι κάτι καινούριο όπως προαναφέρθηκε. Λόγοι όπως έλλειψη τοπικού αποθηκευτικού χώρου, η ανάγκη κοι-

νοχρησίας πληροφοριών ή η ανάγκη για πρόσβαση στις αποθηκευμένες πληροφορίες ανά πάσα στιγμή οδήγησαν στην δημιουργία των λεγόμενων υπηρεσιών αποθήκευσης στο νέφους (cloud storage). Το πρώτο ολοκληρωμένο σύστημα αποθήκευσης δεδομένων στο διαδίκτυο, με την σύγχρονη έννοια του όρου, είναι το Amazon S3 ή αλλιώς Amazon Simple Storage Service [?] το οποίο ξεκίνησε την λειτουργία του το 2006 . Ακολούθησαν και άλλες αντίστοιχες πλατφόρμες όπως το Dropbox [?], το Google Cloud Storage [?]. Ανάμεσα στις πλατφόρμες διακρίνονται διαφορές όσον αφορά τον τρόπο αποθήκευσης, την διαχείριση των δεδομένων κ.α. Κοινός στόχος όλων είναι η διατήρηση των δεδομένων που αποστέλλει ένας χρήστης και η επαναπροώθηση τους στον ίδιο ή σε όποιον άλλο χρήστη της πλατφόρμας επιλέξει ο ιδιοκτήτης των δεδομένων. Σε μια τέτοιου είδους υπηρεσία είναι λογικό να απαιτούνται και εγγυήσεις ασφάλειας για τα δεδομένα. Οι εγγυήσεις αυτές αφορούν τόσο την αποθήκευση στα μηχανήματα που αποτελούν το υπολογιστικό νέφος όσο και την διαδικασία με την οποία διαμοιράζονται οι πληροφορίες στους κατάλληλους χρήστες.

Κοινό χαρακτηριστικό των παραπάνω υπηρεσιών είναι ότι ο εκάστοτε χρήστης κάνει εγγραφή στην υπηρεσία και στην συνέχεια μεταφορτώνει τα δεδομένα που θέλει και αυτά αποθηκεύονται στην υπηρεσία. Ο χρήστης δεν έχει γνώση για τον τρόπο, το μέσο ή την τοποθεσία αποθήκευσης των δεδομένων του. Ο τρόπος που μπορεί να επιτευχθεί το απαιτούμενο επίπεδο ασφάλειας είναι η κρυπτογράφηση των δεδομένων. Οι δύο πιο διαδεδομένοι και ευρέως αποδεκτοί τύποι κρυπτογράφησης είναι η συμμετρική και η κρυπτογράφηση δημόσιου κλειδιού. Πλέον, υπάρχουν και πιο σύγχρονοι τρόποι όπως η κρυπτογράφηση με βάση κάποιο χαρακτηριστικό ή η ομομορφική κρυπτογράφηση. Αυτοί οι τύποι κρυπτογραφήσεων θα παρουσιαστούν στην συνέχεια.

2.2.1 Συμμετρική κρυπτογράφηση

Στην συμμετρική κρυπτογράφηση υπάρχει μοναδικό μυστικό κλειδί το οποίο χρησιμοποιείται για τις διαδικασίες της κρυπτογράφησης και της αποκρυπτογράφησης. Οι οντότητες που επικοινωνούν με τον συγκεκριμένο τρόπο θα πρέπει πρώτα να έχουν ανταλλάξει αυτό το κλειδί ώστε να βρίσκεται στην κατοχή και των δύο πλευρών. Βασικό πλεονέκτημα αυτού του είδους κρυπτογράφησης είναι η υψηλή απόδοση που μπορεί να επιτύχει. Λόγω αυτού του πλεονεκτήματος χρησιμοποιεί-

ται για την κρυπτογράφηση μεγάλου όγκου δεδομένων όπως για παράδειγμα τα περιεχόμενα μιας βάσης δεδομένων. Υπάρχουν δύο τύποι αλγορίθμων συμμετρικής κρυπτογράφησης. Οι αλγόριθμοι που χρησιμοποιούν blocks και οι αλγόριθμοι ροής (Stream algorithms). Τα πιο δημοφιλή παραδείγματα αλγορίθμων συμμετρικής κρυπτογράφησης είναι ο Advanced Encryption Standard (AES) καθώς και ο blowfish.

2.2.2 Οι πέντε βασικές παραλλαγές του αλγορίθμου AES

1. ECB mode (Electronic Code Book mode): Αποτελεί τον πιο απλοϊκό τρόπο κρυπτογράφησης AES. Διαχωρίζει τα δεδομένα σε blocks ίσου μεγέθους με τα block του AES. Έπειτα κάθε block κρυπτογραφείται με το ίδιο ακριβώς κλειδί.

2. CBC (Cipher Block Chaining): Ο συγκεκριμένος τρόπος κρυπτογράφησης χρησιμοποιεί διάνυσμα αρχικοποίησης (Initialization vector). Τα αρχικά δεδομένα διασπώνται σε blocks. Πραγματοποιείται η λογική πράξη xor μεταξύ του πρώτου plaintext block με το διάνυσμα αρχικοποίησης. Στην συνέχεια γίνεται κρυπτογράφηση του αποτελέσματος που προέκυψε με το συμμετρικό κλειδί. Ακολούθως, πραγματοποιείται λογική πράξη xor του δεύτερου plaintext block με το αποτέλεσμα της προηγούμενης κρυπτογράφησης κ.ο.κ.

3. CFB (Cipher FeedBack): Επίσης χρησιμοποιεί διάνυσμα αρχικοποίησης. Αρχικά γίνεται συμμετρική κρυπτογράφηση του διανύσματος. Στην συνέχεια πραγματοποιείται λογική πράξη xor με το πρώτο plaintext block. Το αποτέλεσμα της πράξης κρυπτογραφείται συμμετρικά και στην συνέχεια πραγματοποιείται λογική πράξη xor με το δεύτερο plaintext block κ.ο.κ.

4. OFB (Output FeedBack): Επίσης χρησιμοποιεί διάνυσμα αρχικοποίησης το οποίο στην αρχή κρυπτογραφείται συμμετρικά. Στην συνέχεια το αποτέλεσμα της κρυπτογράφησης χρησιμοποιείται δύο φορές. Αρχικά, ως είσοδος μαζί με το πρώτο plaintext block σε λογική πράξη xor και επιπλέον ως είσοδος για κρυπτογράφηση με συμμετρικό κλειδί. Το αποτέλεσμα της τελευταίας κρυπτογράφησης χρησιμοποιείται αντίστοιχα δύο φορές όπως πριν κ.ο.κ.

5. CTR mode: Ο συγκεκριμένος τρόπος χρησιμοποιείται στην παρούσα σχεδίαση. Αρχικοποιείται μετρητής ο οποίος κρυπτογραφείται συμμετρικά. Στην

συνέχεια πραγματοποιείται λογική πράξη xor με το αποτέλεσμα της κρυπτογράφησης και του πρώτου plaintext block. Ο μετρητής αυξάνεται κατά ένα και κρυπτογραφείται. Αντίστοιχα, πραγματοποιείται λογική πράξη xor του αποτελέσματος της προηγούμενης κρυπτογράφησης με το δεύτερο plaintext block κ.ο.κ.

2.2.3 Ασύμμετρη κρυπτογράφηση / Κρυπτογράφηση Δημόσιου – Ιδιωτικού κλειδιού

Ο δεύτερος τύπος κρυπτογράφησης ονομάζεται ασύμμετρη κρυπτογράφηση ή αλλιώς κρυπτογράφηση δημοσίου ιδιωτικού κλειδιού. Στο συγκεκριμένο τύπο το κλειδί διαχωρίζεται σε δύο ξεχωριστά μέρη, το δημόσιο και το ιδιωτικό. Το δημόσιο κλειδί πρέπει να είναι προσβάσιμο από οποιονδήποτε ενώ το ιδιωτικό θα πρέπει να το γνωρίζει μόνο ο κάτοχος του. Το δημόσιο κλειδί χρησιμοποιείται για την κρυπτογράφηση των δεδομένων καθώς και για τον έλεγχο κάποιας ψηφιακής υπογραφής. Το ιδιωτικό κλειδί χρησιμοποιείται ώστε να αποκρυπτογραφηθούν δεδομένα που κάποιος έχει κρυπτογραφήσει με το αντίστοιχο δημόσιο. Η χρήση αυτού του είδους κρυπτογράφησης είναι διαδεδομένη στην ανταλλαγή μηνυμάτων, αρχείων και οποιονδήποτε άλλων δεδομένων ανάμεσα σε δύο χρήστες. Ο πρώτος δημοφιλής αλγόριθμος ανταλλαγής κλειδιού ήταν ο αλγόριθμος Diffie-Hellman ο οποίος δημοσιοποιήθηκε το 1976 [?].

2.2.4 Ασφαλής επικοινωνία με χρήση TLS (Transport Layer Security)

Το TLS είναι ένα πρωτόκολλο που προσφέρει ασφαλή επικοινωνία μεταξύ ενός διακομιστή και φυλλομετρητών. Το συγκεκριμένο πρωτόκολλο χρησιμοποιείται και στην παρούσα σχεδίαση. Το TLS αποτελεί μετεξέλιξη του Secure Sockets Layer (SSL). Παρακάτω παρουσιάζονται τα βήματα που ακολουθεί το πρωτόκολλο.

1. Ο φυλλομετρητής συνδέεται με τον διακομιστή με https και ζητάει την ταυτότητα του διακομιστή.
2. Ο διακομιστής στέλνει αντίγραφο πιστοποιητικού που περιέχει το δημόσιο κλειδί του.

3. Ο φυλλομετρητής ελέγχει την εγκυρότητα του πιστοποιητικού ρίζας, δημιουργεί συμμετρικό κλειδί συνεδρίας, κρυπτογραφεί το κλειδί συνεδρίας με το δημόσιο κλειδί του διακομιστή και το στέλνει πίσω.

4. Ο διακομιστής αποκρυπτογραφεί το συμμετρικό κλειδί συνεδρίας χρησιμοποιώντας το ιδιωτικό κλειδί του και στέλνει πίσω επιβεβαίωση κρυπτογραφημένη με κλειδί συνεδρίας.

5. Ο διακομιστής και ο φυλλομετρητής κρυπτογραφούν όλη την ανταλλασσόμενη πληροφορία με το κλειδί συνεδρίας.

2.3 Κρυπτογράφηση με βάση κάποιο χαρακτηριστικό

Η κρυπτογράφηση με βάση κάποιο χαρακτηριστικό (Attribute-Based Encryption) των δεδομένων είναι μία σύγχρονη τεχνική κρυπτογράφησης η οποία έχει απασχολήσει την ερευνητική κοινότητα τα τελευταία χρόνια και πάνω στην οποία γίνεται συστηματική έρευνα. Λόγω του ότι αποτελεί μία αρκετά καινούργια μορφή κρυπτογράφησης δεν έχει χρησιμοποιηθεί ακόμη σε εύρος σε υπαρκτά συστήματα. Αρχικά θα πρέπει να παρουσιαστεί το πρόβλημα το οποίο προσπαθεί να λύσει αυτού του είδους η κρυπτογράφηση. Έστω λοιπόν ότι κάποιος θέλει να μοιραστεί τις πληροφορίες που διαθέτει με μία ομάδα ατόμων όπως οι εργαζόμενοι μιας συγκεκριμένης υπηρεσίας. Κοινό χαρακτηριστικό που έχουν όλοι οι εργαζόμενοι είναι ότι ανήκουν στην ίδια ακριβώς υπηρεσία. Επομένως η αποκρυπτογράφηση θα πρέπει να μπορεί να γίνει μόνο από τα συγκεκριμένα άτομα που διαθέτουν αυτό το μοναδικό χαρακτηριστικό σε σχέση με άλλες οντότητες. Συμπερασματικά δηλαδή εφαρμόζεται ένας έλεγχος πρόσβασης στα δεδομένα.

Αυτού του είδους η κρυπτογράφηση λοιπόν βασίζεται στο κοινό χαρακτηριστικό των ατόμων το οποίο στην προκειμένη περίπτωση είναι η συμμετοχή σε κοινή υπηρεσία. Επομένως, δημιουργείται μία πολιτική πρόσβασης στα δεδομένα. Αυτός ο τρόπος ίσως να είναι και πιο εύκολα κατανοητός για τους απλούς χρήστες καθώς πλέον η πρόσβαση στα δεδομένα καθορίζεται από κάποιο χαρακτηριστικό όπως η ταυτότητα η περιοχή στην οποία βρίσκονται και όχι από κάποιο ιδιωτικό κλειδί.

Επομένως, η βασική δυνατότητα που προσφέρει αυτός ο τρόπος κρυπτογράφησης είναι η επιλογή του χαρακτηριστικού εκείνου το οποίο διαχωρίζει τους χρήστες

μιας υπηρεσίας με τον επιθυμητό τρόπο. Επιπροσθέτως, επειδή οι πληροφορίες κρυπτογραφούνται με βάση κάποιο χαρακτηριστικό δεν απαιτείται να υπάρχει κάποια κεντρική αρχή η οποία θα αποφασίσει για τον έλεγχο πρόσβασης στα δεδομένα. Συνεπώς, δεν απαιτείται ο χρήστης να έχει εμπιστοσύνη στον πάροχο για να ελέγχει σε ποιον θα διοχετευθούν τα δεδομένα. Η πρώτη ολοκληρωμένη δουλειά πάνω στο συγκεκριμένο τύπο κρυπτογράφησης έγινε το 2005 από τους Sahai και Waters οι οποίοι εισήγαγαν τον όρο Fuzzy identity based encryption [?]. Η συγκεκριμένη εργασία είναι αυτή πάνω στην οποία στηρίζονται κατά κύριο λόγο όλες οι επιστημονικές έρευνες πάνω στο συγκεκριμένο τρόπο κρυπτογράφησης. Μάλιστα, προτείνεται κρυπτογράφηση με βάση βιομετρικά χαρακτηριστικά των χρηστών όπως ας πούμε σάρωση της ίριδος του ματιού του χρήστη με σκοπό την απόκτηση πρόσβασης στο αρχικό κείμενο. Ένα ακόμα χαρακτηριστικό μπορεί να είναι η θέση που έχει ένας εργαζόμενος σε μια εταιρεία. Παρακάτω θα δούμε τις βασικές λειτουργίες που απαιτούνται [?].

1. Αρχικοποίηση: Ο αλγόριθμος αρχικοποίησης ο οποίος δέχεται ως είσοδο την παράμετρο ασφάλειας και την περιγραφή του συνόλου χαρακτηριστικών. Η έξοδος της είναι οι δημόσιες παράμετροι και ένα master κλειδί.

2. Κρυπτογράφηση: Ο αλγόριθμος κρυπτογράφησης δέχεται ως είσοδο τις δημόσιες παραμέτρους το μήνυμα M και την δομή πρόσβασης A. Ο αλγόριθμος κρυπτογραφεί το M με βάση τις δημόσιες παραμέτρους και παράγει το κρυπτογραφημένο κείμενο.

3. Δημιουργία Κλειδιού: Ο αλγόριθμος παραγωγής κλειδιού λαμβάνει ως είσοδο το master κλειδί και μια ομάδα χαρακτηριστικών που περιγράφουν το κλειδί. Ως έξοδος παράγεται το ιδιωτικό κλειδί.

4. Αποκρυπτογράφηση: Ο αλγόριθμος αποκρυπτογράφησης δέχεται ως είσοδο τις δημόσιες παραμέτρους, το κρυπτογραφημένο κείμενο, το οποίο περιέχει την πολιτική πρόσβασης και το ιδιωτικό κλειδί για μια ομάδα χαρακτηριστικών. Αν ικανοποιούνται οι απαραίτητες συνθήκες το πραγματοποιείται η αποκρυπτογράφηση και προκύπτει εκ νέου το αρχικό μήνυμα.

2.4 Ομομορφική κρυπτογράφηση

Οι σύγχρονοι αλγόριθμοι κρυπτογράφησης όπως αυτοί που παρουσιάστηκαν στις προηγούμενες ενότητες θεωρούνται επαρκώς ασφαλείς καθώς απαιτείται πολύ υψηλή υπολογιστική ισχύς και ικανότητα ώστε να είναι δυνατή η παράκαμψη τους σε ένα εύλογο χρονικό διάστημα. Όμως, ένα μειονέκτημα τους είναι ότι για να επεξεργαστεί κάποιος τα δεδομένα που έχουν κρυπτογραφηθεί θα πρέπει πρώτα αυτά να αποκρυπτογραφηθούν. Σε μία τέτοια περίπτωση τα δεδομένα θα βρεθούν, έστω και για και για ένα ελάχιστο χρονικό διάστημα, στην αρχική τους μορφή γεγονός που τα καθιστά ευάλωτα απέναντι σε κάποια επίθεση. Το αποτέλεσμα μιας τέτοιας επίθεσης θα μπορούσε να είναι καταστροφικό. Επομένως απαιτείται ένας τρόπος επεξεργασίας των δεδομένων ο οποίος δεν θα θέτει σε κίνδυνο την εμπιστευτικότητα καθώς και την ακεραιότητα τους. Λύση στο πρόβλημα που προαναφέρθηκε αποτελεί η ομομορφική κρυπτογράφηση. [?].

Η ομομορφική κρυπτογράφηση αποτελεί μία μέθοδο κρυπτογράφησης στην οποία τα δεδομένα παραμένουν κρυπτογραφημένα κατά τη διάρκεια της επεξεργασίας τους. Επιτρέπει δηλαδή μερικές σύνθετες μαθηματικές διαδικασίες να εφαρμοστούν πάνω στα κρυπτογραφημένα δεδομένα χωρίς να τίθεται σε κίνδυνο η εμπιστευτικότητα τους. Φυσικά, το αποτέλεσμα αυτών των μαθηματικών διαδικασιών είναι το ίδιο με το αποτέλεσμα που θα προέκυπτε αν οι διαδικασίες αυτές εφαρμόζονταν πάνω στα αρχικά, μη κρυπτογραφημένα, δεδομένα. Η ομομορφική κρυπτογράφηση μπορεί να χρησιμοποιηθεί σε περιπτώσεις όπως η αποθήκευση ευαίσθητων δεδομένων στο νέφος. Σε αυτήν την περίπτωση μπορεί οι χρήστες του νέφους να μην εμπιστεύονται τον πάροχο σε βαθμό που θα του επιτρέπουν να έχει πρόσβαση στην αρχική μορφή των δεδομένων τους. Εμπιστεύονται όμως τον πάροχο να εφαρμόσει τις μαθηματικές διαδικασίες. Σε μία τέτοια περίπτωση λοιπόν τα δεδομένα παραμένουν ασφαλή και μπορεί να πραγματοποιηθεί με ασφάλεια η επεξεργασία τους από τον ίδιο τον πάροχο. Ένα παράδειγμα στο οποίο είναι χρήσιμη η εφαρμογή της ομομορφικής κρυπτογράφησης είναι αυτό της αποθήκευσης βιομετρικών δεδομένων πολιτών. Συχνά απαιτείται επεξεργασία ή κάποια ενημέρωση τους χωρίς φυσικά να τίθεται σε κίνδυνο η εμπιστευτικότητα των δεδομένων. Στην προκειμένη περίπτωση μπορεί να γίνει επεξεργασία των δεδομένων ενώ εκείνα παραμένουν σε κρυπτογραφημένη μορφή. Αυτό το σενάριο είναι και το πιο χαρακτηριστικό παράδειγμα στο οποίο η ομομορφική κρυπτογράφηση μπορεί να δώσει την λύση που άλλες μορφές

κρυπτογράφησης δεν μπορούν. Ένα τελευταίο παράδειγμα είναι η εξαγωγή συμπερασμάτων από δεδομένα τα οποία είναι ευαίσθητα. Δηλαδή, ενώ απαιτείται να μπορεί κάποιος να αξιοποιήσει τα δεδομένα, αυτά θα πρέπει παραμένουν κρυφά.

Στόχος της πλήρους ομομορφικής κρυπτογράφησης είναι να δώσει τη δυνατότητα σε οποιονδήποτε να χρησιμοποιήσει κρυπτογραφημένα δεδομένα ώστε να πραγματοποιήσει χρήσιμες λειτουργίες. Φυσικά, ο ενδιαφερόμενος μπορεί να έχει πρόσβαση σε όλα τα δεδομένα. Επιπλέον, μπορούν να συνυπάρχουν πάνω από μία οντότητες οι οποίες έχουν πρόσβαση στα δεδομένα χωρίς αυτό να θέτει σε ρίσκο την ασφάλεια τους ή του συστήματος. Μπορούν να πραγματοποιηθούν μαθηματικές πράξεις όπως η πρόσθεση ή ο πολλαπλασιασμός για έναν απεριόριστο αριθμό επαναλήψεων. Ένα παράδειγμα στο οποίο αυτός ο τύπος κρυπτογράφησης μπορεί να καταστεί λειτουργικός είναι η αποθήκευση δεδομένων στο νέφος. Ένα σημαντικό μειονέκτημα όμως είναι η ταχύτητα σε σχέση με την πραγματοποίηση των μαθηματικών πράξεων πάνω σε δεδομένα τα οποία δεν είναι κρυπτογραφημένα. Έχει σημειωθεί αξιοσημείωτη πρόοδος στον τομέα της απόδοσης. Για παράδειγμα η εταιρεία Microsoft χρησιμοποιεί ομομορφική κρυπτογράφηση για την διαχείριση δεδομένων υγειονομικού ή οικονομικού ενδιαφέροντος [?] ωστόσο παραμένει ένα ανοιχτό ζήτημα για έρευνα και δοκιμή. Συμπερασματικά, η ομομορφική κρυπτογράφηση είναι ένας τομέας του χώρου της ιδιωτικότητας των δεδομένων ο οποίος έχει θετικές προοπτικές και πάνω στον οποίο υπάρχει, ειδικά τα τελευταία χρόνια, συστηματική έρευνα. Αν και ακόμα η απόδοση υλοποιήσεων που χρησιμοποιούν αυτόν τον τύπο κρυπτογράφησης δεν έχουν γενικά αποδεκτή απόδοση με την πρόοδο που αναμένεται να γίνει σε αυτόν τον τομέα θα υπάρχουν στο μέλλον λύσεις σε πραγματικά προβλήματα.

2.4.1 Μαθηματικό υπόβαθρο πλήρους ομομορφικής κρυπτογράφησης

Όπως αναφέρθηκε η πλήρως ομομορφική κρυπτογράφηση μπορεί να υποστηρίξει τις πράξεις της πρόσθεσης και του πολλαπλασιασμού. Στα μαθηματικά ένας δακτύλιος ορίζεται ως μια αβελιανή ομάδα R. Μια αβελιανή ομάδα είναι μια ομάδα στην οποία το αποτέλεσμα της πράξης μεταξύ δύο στοιχείων της ομάδας δεν επηρεάζεται από την σειρά με την οποία τοποθετούνται στην πράξη. Η ομάδα R διαθέτει τις πράξεις της πρόσθεσης και του πολλαπλασιασμού και ακολουθά τα παρακάτω 8 αξιώματα τα οποία ονομάζονται τα αξιώματα δακτυλίου [?].

1. Η πράξη της πρόσθεσης υποστηρίζει την προσεταιριστική ιδιότητα.
2. Το 0 είναι ουδέτερο στοιχείο της πρόσθεσης.
3. Το -α είναι το προσθετικό αντίθετο του α.
4. Η πράξη της πρόσθεσης υποστηρίζει την αντιμεταθετική ιδιότητα.

Το R είναι μονοειδές ως προς τον πολλαπλασιασμό το οποίο σημαίνει πως:

5. Η πράξη του πολλαπλασιασμού υποστηρίζει την προσεταιριστική ιδιότητα.
6. Το α είναι το ουδέτερο στοιχείο της πρόσθεσης.

Η πράξη του πολλαπλασιασμού μπορεί να κατανεμηθεί πάνω στην πράξη της πρόσθεσης.

7. $\alpha \cdot (\beta + \gamma) = (\alpha \cdot \beta) + (\alpha \cdot \gamma)$, $\forall$ α, β, γ στο R (αριστερή κατανομή).
8. $(\beta+\gamma) \cdot \alpha = (\beta \cdot \alpha) + (\gamma \cdot \alpha)$, $\forall$ α, β, γ στο R (δεξιά κατανομή).

Έστω ότι R και S είναι δύο δακτύλιοι, τότε ένας δακτύλιος ομομορφισμού είναι μια συνάρτηση:

$$f : R \Rightarrow S$$

τέτοια ώστε:

$$f(\alpha + \beta) = f(\alpha) + f(\beta)$$

$$f(\alpha \cdot \beta) = f(\alpha) \cdot f(\beta)$$

$\forall$ α, β στο R.

Έστω (P, C, K, E, D) ένα σχήμα κρυπτογράφησης όπου P, C είναι τα αρχικό και το κρυπτογραφημένο κείμενο αντίστοιχα, το K είναι το κλειδί και τα E και D είναι οι αλγόριθμοι κρυπτογράφησης και αποκρυπτογράφησης αντίστοιχα. Υποθέτουμε ότι τα αρχικά κείμενα σχηματίζουν ένα δακτύλιο (P,$\oplus p$,$\otimes p$) και τα κρυπτογραφημένα κείμενα σχηματίζουν ένα δακτύλιο (C,$\oplus c$,$\otimes c$) τότε ο αλγόριθμος κρυπτογράφησης E είναι μια απεικόνιση από το δακτύλιο P στον δακτύλιο C τέτοιος ώστε E_k : P $\rightarrow$ C, όπου k $\in$ K είναι, είτε το μυστικό κλειδί (σε ένα κρυπτοσύστημα μυστικού κλειδιού), είτε το δημόσιο κλειδί (σε ένα κρυπτοσύστημα δημόσιου κλειδιού).
Για όλα τα a και b στο P και k στο K, αν ισχύει:

$$E_k(a) \oplus_c E_k(b) = E_k(a \oplus_p b)$$
$$E_k(a) \otimes_c E_k(b) = E_k(a \otimes_p b)$$

Τότε το σχήμα κρυπτογράφησης είναι πλήρως ομομορφικό.

2.5 Επανακρυπτογράφηση δεδομένων

Όπως αναφέρθηκε και στην εισαγωγή, η επανακρυπτογράφηση των δεδομένων προσφέρει ένα ακόμα επίπεδο ασφάλειας. Με τον όρο επανακρυπτογράφηση δεδομένων εννοείται η αλλαγή του κλειδιού κρυπτογράφησης πάνω στα ίδια δεδομένα. Δηλαδή, τα δεδομένα αποκρυπτογραφούνται και στην συνέχεια κρυπτογραφούνται με ένα νέο κλειδί. Πρόκειται για μια τεχνική η οποία έχει προταθεί και συνεχώς γίνεται έρευνα με σκοπό την βελτίωση της απόδοσης καθώς όπως είναι κατανοητό μια τέτοια τεχνική αυξάνει το υπολογιστικό κόστος το οποίο έχει μια πλατφόρμα η οποία την εφαρμόζει. Μια ενδιαφέρουσα εργασία που χρησιμοποιεί επανακρυπτογράφηση δεδομένων παρουσιάστηκε από τους Syalim, Nishide και Shakurai [?]. Στην συγκεκριμένη εργασία πρώτα πραγματοποιείται μετατροπή των δεδομένων με μια μετατροπή All or Nothing Transform (AONT) η οποία γενικότερα έχει μερικά αξιοπρόσεκτα χαρακτηριστικά. Αυτά είναι τα εξής:

- Το αποτέλεσμα της μετατροπής είναι ψευδοτυχαίο οπότε η πιθανότητα να παραχθεί η ίδια έξοδος για δύο block δεδομένων είναι πολύ μικρή.
- Το αποτέλεσμα της μετατροπής μπορεί να αναστραφεί στο αρχικό μόνο αν όλα τα block βρίσκονται στην σωστή θέση.
- Το αποτέλεσμα της μετατροπής δεν μπορεί να αναστραφεί στο αρχικό αν κάποιο κομμάτι λείπει.

Στην συνέχεια μπορεί να εφαρμοστεί συμμετρική επανακρυπτογράφηση των δεδομένων. Η εργασία έπειτα μετεξελίχθηκε [?] με σκοπό το σύστημα που σχεδιάστηκε να είναι ασφαλές απέναντι σε επιθέσεις Chosen Plaintext Attack (CPA). Σε αυτό το είδος επίθεσης ο επιτιθέμενος στέλνει διαρκώς δεδομένα για κρυπτογράφησης και στην συνέχεια παραλαμβάνει τα δεδομένα σε κρυπτογραφημένη μορφή.

Ακολούθως, εξετάζει τα κρυπτογραφημένα δεδομένα σε σχέση με τα αρχικά που έστειλε με σκοπό να εξάγει ολόκληρο ή ένα μέρος του κλειδιού. Επομένως, για να αντιμετωπιστεί αυτή η επίθεση χρησιμοποιήθηκε ένας νέος τύπος All on Nothing Transformation στον οποίο κάθε block δεδομένων κρυπτογραφείται με ένα τυχαίο κλειδί. Ακολούθως, πραγματοποιείται πράξη xor μεταξύ του τυχαίου κλειδιού και του hash όλων των blocks έτσι ώστε να μην μπορεί να γίνει μερική ανάκτηση του κλειδιού κρυπτογράφησης χωρίς κάποιος να έχει όλα τα blocks. Με αυτόν τον τρόπο θα πρέπει κανείς να έχει όλα τα δεδομένα για να τα αποκρυπτογραφήσει και όχι μόνο ένα τμήμα τους. Αυτό το γεγονός αυξάνει λοιπόν τον βαθμό δυσκολίας μιας τέτοιας επίθεσης.

2.6 Trusted Platform module (TPM)

Για δεκαετίες η κρυπτογράφηση των δεδομένων γινόταν μέσω κατάλληλων αλγορίθμων τους οποίους αναλάμβανε το λογισμικό να υλοποιήσει . Για μεγάλο χρονικό διάστημα αυτό ήταν επαρκές και κάλυπτε τις απαιτήσεις ασφαλείας που είχαν πραγματικές εφαρμογές. Όμως, υπήρξαν επιθέσεις με σκοπό την παραβίαση ευαίσθητων δεδομένων των πολιτών όπως για παράδειγμα είναι τα ιατρικά ιστορικά. Πιο συγκεκριμένα, υπήρξαν 58 καταγεγραμμένες επιθέσεις στις ΗΠΑ του 2019 που οδήγησαν στην παραβίαση 1.4 εκατομμυρίων ιατρικών καρτελών πολιτών [?]. Επομένως, η αύξηση της συχνότητας καθώς και της πολυπλοκότητας των επιθέσεων επέβαλε την έρευνα για ανακάλυψη νέων λύσεων. Συνεπώς, πέρα από τη βελτίωση των λύσεων που βασίζονται στο λογισμικό υπήρξε και η ανάγκη για λύσεις οι οποίες χρησιμοποιούν ειδικό υλικό. Υπάρχει κοινή προσπάθεια από διάφορες ιδιωτικές εταιρείες διεθνούς φήμης για την παροχή και την εξέλιξη αυτής της τεχνολογίας. Για τον κοινό αυτό σκοπό δημιουργήθηκε η ένωση trusted computing group η οποία απαρτίζεται από μέλη όπως η Dell, η Intel, η Hewlett-Packard, η Cisco, η IBM κ.α. [?]

Αυτές οι λύσεις χρησιμοποιούν ειδικά ολοκληρωμένα κυκλώματα τα οποία ονομάζονται trusted platform modules ή εν συντομία TPMs. Τέτοια ολοκληρωμένα κυκλώματα μπορούν να αξιοποιηθούν από όλους τους τύπους υπολογιστών όπως υπολογιστές ιδιωτικής χρήσης ή διακομιστές. Επιπλέον μπορούν να χρησιμοποιηθούν και από κινητές συσκευές. Τα TPMs πραγματοποιούν ενέργειες όπως η παραγωγή

τυχαίων αριθμών, η παραγωγή κρυπτογραφικών κλειδιών, η ασφαλής αποθήκευση δεδομένων και η απομακρυσμένη επιβεβαίωση. Μάλιστα, παρέχονται εγγυήσεις για την αντοχή ακόμη και απέναντι σε φυσικές επιθέσεις. Οι πληροφορίες που αποθηκεύονται στο υλικό είναι καλύτερα προστατευμένες από εξωτερικές επιθέσεις λογισμικού. Έχουν υλοποιηθεί διάφορες εφαρμογές οι οποίες αποθηκεύουν ευαίσθητες πληροφορίες με τη βοήθεια TPMs. Τέτοιες εφαρμογές καθιστούν την μη εξουσιοδοτημένη πρόσβαση πληροφοριών στις συσκευές πολύ δυσκολότερη. Σε περίπτωση όπου ρυθμίσεις της λειτουργίας μιας πλατφόρμας που βασίζεται πάνω σε αυτή την τεχνολογία μεταβληθούν τότε η πρόσβαση στα ευαίσθητα δεδομένα αποκόπτεται. Μία σημαντική παρατήρηση είναι πώς τα TPMs δεν είναι σε θέση να ελέγξουν το λογισμικό το οποίο εκτελείται στο σύστημα.

Μία χρήσιμη εφαρμογή της συγκεκριμένης τεχνολογίας είναι να χρησιμοποιηθεί στο υπολογιστικό νέφος. Για παράδειγμα, αν κατά τη διάρκεια της εκκίνησης ενός υπολογιστή ανιχνευθούν απρόσμενες αλλαγές στις ρυθμίσεις του συστήματος τότε αυτόματα οι εφαρμογές που έχουν υψηλές απαιτήσεις σε ασφάλεια μπορούν να σταματήσουν να λειτουργούν έως ότου αποκατασταθούν οι αλλαγές. Η διαδικασία αυτή παρέχει ένα ακόμα επίπεδο ασφάλειας στο σύστημα καθώς και την εγγύηση στους χρήστες ότι το σύστημα λειτουργεί σύμφωνα με τις προβλεπόμενες και απαραίτητες προϋποθέσεις. Μια τέτοιου είδους εγγύηση έχει ακόμα μεγαλύτερη σημασία όταν ο κώδικας ο οποίος επιθυμεί ο χρήστης να εκτελεστεί με ασφάλεια εκτελείται απομακρυσμένα σε κάποιο σύστημα υπολογιστικού νέφους.

2.6.1 TPM version 1 (TPM1)

Η πρώτη έκδοση αυτής της τεχνολογίας η οποία χρησιμοποιήθηκε σε μία μεγάλη κλίμακα ήταν η έκδοση 1.1 η οποία εκδόθηκε το 2003. Ακόμα και εκείνη την εποχή βασικές λειτουργίες των TPMs ήταν διαθέσιμες. Αυτές οι λειτουργίες ήταν η παραγωγή κλειδιών, αν και αυτό περιοριζόταν στα κλειδιά για κρυπτογράφηση RSA, ασφαλής αποθήκευση, ασφαλής εξουσιοδότηση, και επιβεβαίωση της ορθής λειτουργίας της συσκευής. Επιπλέον, με σκοπό την παροχή εγγυήσεων ιδιωτικότητας τα TPMs κατασκευάζουν μυστικά κλειδιά τα οποία ονομάζονται κλειδιά ανώνυμης ταυτότητας. Για τον έλεγχο της αυθεντικότητας τέτοιο κλειδιών, καθώς μόνο αληθινά TPMs μπορούν να τα κατασκευάσουν, δημιουργήθηκε μία νέα οντότητα διαδικτύου η οποία ονομάστηκε αρχή ιδιωτικών συμφωνητικών (privacy certificate authority)

και η οποία είναι σε θέση να επιβεβαιώσει ότι ένα τέτοιο κλειδί παρήχθη όντως από ένα αληθινό TPM χωρίς να προσδιορίζει την ταυτότητα αυτού [?]. Αρχικά δεν ήταν στόχος της ένωσης TCG να κάνει την σχεδίαση των TPM ανθεκτική απέναντι σε φυσικές επιθέσεις. Αν και τέτοιες δυνατότητες υπήρχαν αποφασίστηκε να αφεθεί στους κατασκευαστές η ευθύνη για την παροχή τέτοιων εγγυήσεων ασφαλείας. Ωστόσο οποιεσδήποτε επιθέσεις πού βασίζονται στο λογισμικό πρέπει να αποτρέπονται από τον σχεδιασμό των TPM.

Ένα μειονέκτημα της έκδοσης 1.1 ήταν οι διαφορετικές διεπαφές λογισμικού πού είχαν οι εκάστοτε κατασκευαστές γεγονός το οποίο οδηγούσε στην ανάγκη για διαφορετικούς οδηγούς (drivers). Στην έκδοση όμως 1.2 έγιναν οι απαραίτητες βελτιώσεις εξουσιοδότησης ώστε να υπάρχει ενιαία διεπαφή λογισμικού. Επιπροσθέτως, στην έκδοση 1.1 τα κλειδιά ανώνυμης ταυτότητας τα οποία αναφέρθηκαν παραπάνω δεν είχαν προστασία απέναντι σε έναν επιτιθέμενο ο οποίος γνώριζε τον κωδικό εξουσιοδότησης. Φυσικά αυτός ο κωδικός είναι μυστικός αλλά δεν υπήρχε κάτι που να σταματήσει τον επιτιθέμενο από το να δοκιμάζει συνεχώς κωδικούς από αντίστοιχα λεξικά. Η συγκεκριμένη είδηση ονομάζεται επίθεση λεξικού (dictionary attack) και είναι μια επίθεση αρκετά διαδεδομένη με σχετικά χαμηλή πολυπλοκότητα και επίπεδο δυσκολίας υλοποίησης από τον επιτιθέμενο. Για αυτό το λόγο στην έκδοση 1.2 έγιναν οι απαραίτητες αλλαγές ώστε τα TPMs να έχουν προστασία απέναντι σε επιθέσεις λεξικών.

Επιπλέον μία επιλογή που έπρεπε να γίνει από τον οργανισμό TCG ήταν ο κατάλληλος αλγόριθμος κατακερματισμού. Οι διαθέσιμες προτάσεις ήταν δύο. Η πρώτη ήταν ο αλγόριθμος MD5 ο οποίος ήταν και πιο ευρέως διαδεδομένος και η δεύτερη ο αλγόριθμος SHA-1 ο οποίος αν και ήταν ισχυρότερος δεν είχε διαδοθεί τόσο. Επομένως, λόγω των μεγαλύτερων εγγυήσεων ασφάλειας που παρέχει ο αλγόριθμος SHA-1 εν τέλει και επιλέχθηκε. Ωστόσο το 2005 δημοσιεύτηκε η πρώτη σημαντική επίθεση πάνω στον αλγόριθμο SHA-1 [?]. Αυτός ήταν και ο βασικός λόγος που ξεκίνησε ο σχεδιασμός της έκδοσης TPM 2.0 η οποία είναι η πλέον σύγχρονη, ευρέως διαδεδομένη και κοινώς αποδεκτή.

2.6.2 TPM version 2 (TPM 2.0)

Η δεύτερη έκδοση των TPM διαφοροποιείται αισθητά σε σχέση με την πρώτη. Καταρχήν, η προσέγγιση που υπήρξε έμοιαζε περισσότερο με τη δημιουργία μιας βι-

βλιοθήκης. Πλέον, είναι επιτρεπτό στους χρήστες να επιλέγουν διαφορετικές λειτουργικότητες που παρέχει το TPM, διαφορετικά επίπεδα υλοποίησης και διαφορετικά επίπεδα ασφαλείας. Επιπλέον προστέθηκαν νέα χαρακτηριστικά και νέες υλοποιημένες συναρτήσεις οι οποίες δίνουν τη δυνατότητα στον χρήστη να δημιουργήσει νέους αλγόριθμους κρυπτογράφησης όποτε χρειαστεί. Φυσικά και αυτή η νέα έκδοση μπορεί να χρησιμοποιηθεί και σε ιδιωτικούς υπολογιστές καθώς και σε ενσωματωμένα συστήματα. Δίνεται δυνατότητα επιλογής διαφορετικών επιπέδων ασφάλειας αναλόγως με τη συσκευή δίνοντας έτσι ευελιξία στους χρήστες. Στην δεύτερη αυτή έκδοση υπάρχουν πέντε διαφορετικά TPMs τα οποία είναι τα εξής:

- Διακριτό TPM (Discrete TPM):
- Ενσωματωμένο TPM (Integrated TPM)
- TPM υλικολογισμικού (Firmware TPM)
- TPM διαχειριστή (Hypervisor TPM)
- TPMs λογισμικού (Software TPMs)

2.7 Ασφάλεια στις δημοφιλέστερες υπηρεσίες νέφους (Cloud services)

Όπως προαναφέρθηκε εταιρείες όπως η Google και η Amazon παρέχουν διάφορες υπηρεσίες με την χρήση υπολογιστικού νέφους. Οι υπηρεσίες αυτές αφορούν την αποθήκευση δεδομένων όπως το Google drive, η υπηρεσία Amazon S3 ή το Dropbox. Και οι τρεις προαναφερθείσες πλατφόρμες είναι ιδιαίτερα δημοφιλείς με την κάθε μια να έχει δεκάδες εκατομμύρια χρήστες. Η βασική διαφορά είναι πως ενώ το Google drive και το Dropbox αποθηκεύουν αρχεία η πλατφόρμα Amazon S3 παρέχει ένα ολόκληρο σύστημα διαχείρισης τους με αρκετές αποφάσεις που αφορούν την ασφάλεια να εξαρτώνται από τους χρήστες. Ενδιαφέρον έχουν μερικά από τα πρωτόκολλα ασφάλειας που χρησιμοποιούν οι εταιρείες αυτές για να παρέχουν εγγυήσεις ασφάλειας στους χρήστες τους. Αρχικά, η Google [?] διατηρεί τα δεδομένα των χρηστών σε κρυπτογραφημένη μορφή είτε αυτά βρίσκονται στους δίσκους των μηχανημάτων της, είτε μεταφέρονται μεταξύ των κέντρων δεδομένων (Data Centers). Τα δεδομένα απομονώνονται από εκείνα των υπόλοιπων χρηστών ακόμα και

αν βρίσκονται στον ίδιο διακομιστή. Ενδιαφέρον έχει ότι όταν ένας χρήστης διαγράφει τα δεδομένα του από την υπηρεσία αυτά μπορούν να διατηρηθούν στους διακομιστές για έως 180 ημέρες και ότι επιπλέον κάποιοι, έστω και ελάχιστοι στον αριθμό, υπάλληλοι έχουν κάποια πρόσβαση στα δεδομένα.

Η εταιρεία Amazon [?] παρέχει στους χρήστες της έναν καινοτόμο τρόπο ελέγχου των δεδομένων τους. Σκοπός είναι ο χρήστης που είναι ιδιοκτήτης των δεδομένων να επιλέγει ως διαχειριστής τους άλλους χρήστες που θα έχουν πρόσβαση. Η διαδικασία αυτή ονομάζεται AWS Identity and Access Management ή εν συντομία IAM. Με την χρήση του IAM ο χρήστης μπορεί να δώσει πρόσβαση σε ένα άτομο ή έχει την δυνατότητα να δημιουργήσει μια ομάδα ατόμων που να έχουν πρόσβαση στα δεδομένα. Επιπροσθέτως, ο χρήστης μπορεί να καθορίσει τα επίπεδα των δικαιωμάτων που έχουν οι υπόλοιποι χρήστες. Για παράδειγμα, μπορεί να δώσει δικαιώματα διαχειριστή τα οποία είναι ίδια με αυτά του ιδιοκτήτη. Φυσικά, μπορεί να δώσει πιο περιορισμένα δικαιώματα. Ως διαχειριστής, ο ιδιοκτήτης των δεδομένων μπορεί να δημιουργήσει και να διανείμει τους κωδικούς πρόσβασης των άλλων χρηστών. Επίσης, παρέχεται και μηχανισμός ανάλυσης της πρόσβασης στα δεδομένα. Ο ιδιοκτήτης μπορεί ανά πάσα στιγμή να ελέγξει ποιος και πότε είχε πρόσβαση. Έτσι μπορεί ο ιδιοκτήτης να αναπροσαρμόσει τις πολιτικές ασφάλειας που ακολουθεί. Συνεπώς, η ευθύνη για τον έλεγχο πρόσβασης μεταφέρεται από την πλατφόρμα που παρέχει η υπηρεσία στον ίδιο τον χρήστη. Το γεγονός αυτό είναι θετικό για χρήστες που διαθέτουν εμπειρία και είναι σε θέση να προβούν σε ενέργειες και να επιλέξουν τις κατάλληλες πολιτικές ασφάλειας. Όμως, μπορεί αυτή η ελευθερία που παρέχεται στους χρήστες να οδηγήσει κάποιον που δεν είναι ιδιαίτερα εξοικειωμένος και ενημερωμένος σε λάθος επιλογές με συνέπεια την εμφάνιση κενών ασφάλειας.

Μια ενδιαφέρουσα επιλογή που παρέχει η Amazon στους πελάτες είναι η δημιουργία ενός εικονικού ατομικού νέφους (Amazon Virtual Private Cloud) [?]. Η Amazon παρέχει κάποιες ελάχιστες εγγυήσεις ασφάλειας όπως η κρυπτογράφηση και η απομόνωση των δεδομένων και στην συνέχεια δίνει στον χρήστη τον έλεγχο για την πρόσβασης. Μια εργασία η οποία ακολουθεί αντίστοιχη λογική είναι το Guardat [?] το οποίο δίνει την δυνατότητα στους χρήστες να ορίσουν μόνοι τους τις πολιτικές που επιθυμούν με σκοπό την προστασία αρχείων. Μάλιστα μπορεί να εφαρμοστεί διαφορετική πολιτική σε κάθε αρχείο εφόσον αυτό είναι επιθυμητό. Βλέπουμε λοιπόν ότι είναι εφικτό να παρέχονται εγγυήσεις ασφάλειας από τους ίδιους τους χρήστες εφόσον εκείνοι τις δημιουργήσουν.

2.8 Ασφαλές περιβάλλον εκτέλεσης

Σκοπός των τεχνολογιών είναι να παρέχουν ένα ασφαλές περιβάλλον εκτέλεσης για το λογισμικό. Για τον λόγο αυτό δημιουργείται ένα ασφαλές υποσύστημα το οποίο ονομάζεται Ασφαλές Περιβάλλον Εκτέλεσης (Trusted Execution Environment ή εν συντομία TEE). Η ασφαλής λειτουργία επικεντρώνεται στις λειτουργίες που πραγματοποιεί ο επεξεργαστής αλλά δεν περιορίζεται εκεί. Ο κώδικας που τρέχει στο ασφαλές περιβάλλον εκτέλεσης διαχωρίζεται και απομονώνεται από το υπόλοιπο σύστημα. Με τον τρόπο αυτό δεν μπορεί κάποια εξωτερική οντότητα να παρεμβληθεί κατά την διάρκεια εκτέλεσης. Σε ένα σύστημα που δεν εφαρμόζεται κάποια τεχνολογία αυτής της μορφής θα μπορούσε μια κακόβουλη οντότητα να επηρεάσει την ορθή εκτέλεση του κώδικα ώστε να μάθει ποιο θα είναι ή ακόμη και να το αλλοιώσει. Γνωρίζοντας ή αλλοιώνοντας το αποτέλεσμα της εκτέλεσης ενός λογισμικού μπορεί κανείς να λάβει τον έλεγχο κάποιας εφαρμογής ή και ολόκληρου του συστήματος. Είναι αντιληπτό ότι μια τέτοια αλληλουχία γεγονότων μπορεί να είναι καταστροφική αν πρόκειται πχ για ένα σύστημα πλοήγησης ενός αεροπλάνου ή ένα τραπεζικό σύστημα.

Αξίζει να αναφερθεί ότι ένα TEE απομονώνει το έμπιστο λογισμικό από τα υπόλοιπα μέρη του συστήματος, ακόμα και από τα υπόλοιπα ασφαλή περιβάλλοντα. Η υλοποίηση των TEEs διαφέρει ανάλογα με τον κατασκευαστή αλλά οι βασικές αρχές είναι οι ίδιες. Αρχικά εκτελείται κώδικας που είναι προεγκατεστημένος στο σύστημα και θεωρείται εξ αρχής ασφαλής. Ο κώδικας αυτός δημιουργεί τις προϋποθέσεις για να δημιουργηθεί το ασφαλές περιβάλλον που απαιτείται για την εκτέλεση των ευαίσθητων εφαρμογών. Στην συνέχεια θα παρουσιαστούν μερικές βασικές υλοποιήσεις τεχνολογιών που δημιουργούν περιβάλλοντα ασφαλούς εκτέλεσης.

2.8.1 Arm TrustZone

Η πρόταση της εταιρείας Arm στον χώρο της παροχής εγγυήσεων ασφάλειας με χρήση υλικού είναι το Arm TrustZone [?]. Σε έναν πυρήνα λειτουργούν ταυτόχρονα ένα, όπως ονομάζεται, ασφαλές λειτουργικό σύστημα και ένα κανονικό λειτουργικό σύστημα. Το TrustZone παρέχει την δυνατότητα σε μια βιβλιοθήκη ή ακόμη και σε ένα ολόκληρο λειτουργικό σύστημα να τρέξει σε μια ασφαλή περιοχή. Οποιοδήποτε λογισμικό δεν θεωρείται ασφαλές δεν έχει πρόσβαση σε αυτήν την ασφαλή περιοχή

ούτε και στους πόρους αυτής. Επομένως γίνεται διαχωρισμός σε μια ασφαλή περιοχή που ονομάζεται "Ασφαλής Κόσμος" (Secure World) και στην υπόλοιπη περιοχή του συστήματος που ονομάζεται "Κανονικός Κόσμος" (Normal World). Μάλιστα η συγκεκριμένη τεχνολογία έχει ενσωματωθεί και σε κινητές συσκευές οι οποίες διαθέτουν τους αντίστοιχους επεξεργαστές. Στα πλεονεκτήματα της συγκεκριμένης υλοποίησης είναι πως τα λειτουργικά συστήματα γενικού σκοπού δεν χρειάζεται να μεταβάλλουν την σχεδίαση τους ώστε να ενισχυθεί η συνολική ασφάλεια του συστήματος. Επιπλέον, δεν υπάρχει κάποιος περιορισμός στην χρήση της πλήρους ισχύος των πυρήνων. Στα μειονεκτήματα είναι πως μετατοπίζεται στον προγραμματιστή η ευθύνη του διαχωρισμού των επιμέρους μερών της εφαρμογής σε ασφαλή και μη. Δηλαδή, ο προγραμματιστής θα πρέπει να είναι σε θέση να λαμβάνει αποφάσεις σχετικά με την ευαισθησία ορισμένων λειτουργιών της εφαρμογής καθώς δεν μπορεί εκείνη να μεταφορτωθεί ολόκληρη μέσα στο ασφαλές περιβάλλον εκτέλεσης.

2.8.2 AMD Secure Encrypted Virtualization (SEV)

Η πρόταση της εταιρείας AMD ονομάζεται AMD Secure Encrypted Virtualization ή εν συντομία SEV [?]. Η βασική ιδέα είναι η δημιουργία εικονικών μηχανών (VMs) οι οποίες απομονώνονται από τον διαχειριστή (Hypervisor). Ενώ γενικά ένας διαχειριστής θεωρείται έμπιστη οντότητα του συστήματος σε αυτήν την πρόταση της AMD απαιτούνται εικονικές μηχανές ώστε να θεωρείται ασφαλής η εκτέλεση των ευαίσθητων εφαρμογών. Σε κάθε ξεχωριστή εικονική μηχανή δίνεται ένα κλειδί AES με το οποίο κρυπτογραφείται το κομμάτι μνήμης που απαιτείται για την εκτέλεση των λειτουργιών της μηχανής. Το κλειδί παράγεται με τυχαίο τρόπο από ειδικό υλικό και αποθηκεύεται σε ειδικούς καταχωρητές στους οποίους δεν έχει απευθείας πρόσβαση το λογισμικό. Βασικό πλεονέκτημα της συγκεκριμένης υλοποίησης είναι πως παρέχεται προστασία ακόμη και απέναντι στον ίδιο τον διαχειριστή του συστήματος. Ένα μειονέκτημα είναι ότι απαιτείται η δημιουργία και η διαχείριση καινούριας εικονικής μηχανής για κάθε ξεχωριστή ευαίσθητη εφαρμογή.

2.8.3 Η τεχνολογία που χρησιμοποιούμε στην παρούσα υλοποίηση, το Intel SGX

Στην παρούσα εργασία κάνουμε χρήση της τεχνολογίας που παρέχει η Intel και ονομάζεται Intel Secure Guard Extensions ή εν συντομία SGX. Η συγκεκριμένη τε-

χνολογία επιτρέπει στο λογισμικό να δημιουργήσει ένα ασφαλές μέρος στην μνήμη το οποίο ονομάζει enclave. Ο κώδικας που εκτελείται εντός του enclave έχει την δυνατότητα να απομονωθεί αποτελεσματικά από τις υπόλοιπες εφαρμογές, το λειτουργικό σύστημα καθώς και τον διαχειριστή του συστήματος. Η εφαρμογή διαχωρίζεται σε δύο μέρη. Αυτό που θεωρείται ευαίσθητο καθώς και αυτό που θεωρείται μη ευαίσθητο. Στην συνέχεια όπως προαναφέρθηκε το λογισμικό δημιουργεί το enclave στο προστατευμένο κομμάτι της μνήμης. Όταν κληθεί μια συνάρτηση που βρίσκεται εντός του enclave μόνο ο κώδικας που βρίσκεται και αυτός εντός έχει πρόσβαση στα δεδομένα και τους πόρους. Κανένα άλλο λογισμικό δεν έχει πρόσβαση εντός του enclave.

2.8.4 Οι εντολές που υποστηρίζονται από το Intel SGX

Η Intel παρέχει συγκεκριμένες εντολές οι οποίες μπορούν να εκτελεστούν από το σύστημα ή τον χρήστη του συστήματος. Φυσικά αυτές οι εντολές είναι και οι μόνες οι οποίες μπορούν να εκτελεστούν καθώς σε διαφορετική περίπτωση δεν θα μπορούσαν να δοθούν εγγυήσεις ασφάλειας. Συγκεκριμένα, υπάρχουν 18 εντολές που μπορούν να εκτελεστούν. Οι 13 από τον διαχειριστή και οι υπόλοιπες 5 από τον χρήστη. Οι εντολές αφορούν λειτουργίες όπως η δημιουργία και η εκκίνηση ενός enclave και διαχείριση δεδομένων και σελίδων από την πλευρά του διαχειριστή. Αντίστοιχα, από την πλευρά του χρήστη οι εντολές αφορούν την είσοδο και την έξοδο από ένα enclave και την δημιουργία ενός κρυπτογραφικού κλειδιού. Ως σελίδα (page) ορίζεται ένα τμήμα εικονικής μνήμης σταθερού μεγέθους το οποίο είναι και το μικρότερο τμήμα μνήμης που μπορεί να διαχειριστεί το λειτουργικό σύστημα. Οι σελίδες εντάσσονται στον πίνακα σελίδων (page table) ο οποίος αποτελεί την δομή δεδομένων το οποίο αντιστοιχεί τις φυσικές με τις εικονικές διευθύνσεις. Στο σχήμα 2.1 διακρίνονται οι εντολές που υποστηρίζει το Intel SGX.

Εντολές διαχειριστή	Περιγραφή
EADD	Προσθήκη σελίδας
EBLOCK	Περιορισμός μιας σελίδας της EPC
ECREATE	Δημιουργία ενός enclave
EDBGRD	Ανάγνωση δεδομένων από τον εκσφαλματωτή
EBDGWR	Εγγραφή δεδομένων από τον εκσφαλματωτή
EINIT	Αρχικοποίηση ενός enclave
ELDB	Φόρτωση μιας EPC σελίδας ως περιορισμένη
ELDU	Φόρτωση μιας EPC σελίδας ως μη περιορισμένη
EPA	Προσθήκη ενός πίνακα εκδόσεων
EREMOVE	Αφαίρεση μιας σελίδας από την EPC
ETRACE	Ενεργοποίηση ελέγχων EBLOCK
EWB	Επανεγγραφή/ακύρωση μιας EPC σελίδας
Εντολές χρήστη	**Περιγραφή**
EENTER	Είσοδος σε enclave
EEXIT	Έξοδος από enclave
EGETKEY	Δημιουργία κρυπτογραφικού κλειδιού
EREPORT	Δημιουργία κρυπτογραφικής αναφοράς
ERESUME	Είσοδος εκ νέου στο enclave

Figure 2.1: Οι εντολές που υποστηρίζει το Intel SGX.

2.8.5 Οι δομές δεδομένων του Intel SGX

Απαραίτητες για την ορθή λειτουργία του SGX είναι και ορισμένες δομές δεδομένων. Ορίζονται σαφώς 13 δομές δεδομένων. Οι 8 από αυτές χρησιμοποιούνται για την διαχείριση του enclave, 3 για την διαχείριση σελίδων μνήμης και 2 ακόμα για την διαχείριση πόρων. Οι δομές που υπάρχουν είναι οι εξής:

- SGX Enclave Control Structure (SECS): Αναπαριστά ένα enclave. Συγκρατεί πληροφορίες όπως το αναγνωριστικό (ID) ή το μέγεθος.
- Thread Control Structure (TCS): Κάθε νήμα στο enclave συσχετίζεται με ένα TCS. Συγκρατεί πληροφορίες όπως το σημείο εισόδου και κάποιον δείκτη σε δομή SSA που περιγράφεται στην συνέχεια.
- State State Area (SSA): Όταν συμβεί μια ασύγχρονη έξοδος από τον enclave κατά την διάρκεια λειτουργίας του τότε η κατάσταση του θα αποθηκευτεί στην δομή SSA που διαθέτει το νήμα.

- Page Information (PAGEINFO): Συγκρατεί πληροφορίες σχετικά με τον πίνακα σελίδων του enclave.
- Security Information (SECINFO): Συγκρατεί μεταδεδομένα σχετικά με τις σελίδες του enclave.
- Paging Crypto MetaData (PCMD): Συγκρατεί μεταδεδομένα της διαδικασίας της κρυπτογράφησης. Σε συνδυασμό με την δομή PAGEINFO παρέχει αρκετή πληροφορία στον επεξεργαστή ώστε να πραγματοποιήσει λειτουργίες όπως επαλήθευση και αποκρυπτογράφηση.
- Version Array (VA): Για να αποθηκεύσει με ασφάλεια εκδόσεις από τις σελίδες του EPC το SGX ορίζει μια ειδική δομή που ονομάζει Version Array. Ο πίνακας αυτός διαθέτει 512 θέσεις κάθε μια από τις οποίες περιέχει.
- Enclave Page Cache Map (EPCM): Είναι μια δομή που χρησιμοποιείται από τον επεξεργαστή ώστε να παρακολουθεί τα περιεχόμενα του Enclave Page Cache (EPC). Το EPCM συγκρατεί μια εγγραφή για κάθε σελίδα που βρίσκεται στο EPC. Το λογισμικό δεν έχει πρόσβαση στο EPCM.
- Enclave Signature Structure (SIGSTRUCT): Περιέχει πληροφορίες σχετικά με την υπογραφή του enclave.
- EINIT Token Structure (EINITTOKEN): Συγκρατεί πληροφορίες οι οποίες επιβεβαιώνουν ότι ένα enclave επιτρέπεται να εκκινήσει.
- Report (REPORT): Περιέχει πληροφορίες όπως κάποια ορίσματα του enclave και κάποιο hash.
- Report Target Info (TARGETINFO): Χρησιμοποιείται για την αναγνώριση του enclave.
- Key Request (KEYREQUEST): Χρησιμοποιείται για την επιλογή κατάλληλου κλειδιού και οποιονδήποτε άλλων παραμέτρων απαιτούνται για την παραγωγή του κλειδιού.

2.8.6 Ο τρόπος λειτουργίας του Intel SGX

Η Intel παρουσίασε SGX [?] το 2015 με τη σειρά επεξεργαστών skylake. Στόχος του είναι η προστασία ευαίσθητων δεδομένων απέναντι σε μη έμπιστους χρήστες

ή ακόμα και σε ήδη εκτεθειμένα συστήματα με τη βοήθεια κρυπτογραφικών μηχανισμών που παρέχει ο επεξεργαστής. Έτσι λοιπόν δημιουργούνται τα enclaves τα οποία δεν είναι προσβάσιμα με το υπόλοιπο σύστημα και τα δεδομένα τους αποκρυπτογραφούνται μόνο εντός του επεξεργαστή. Με τον τρόπο αυτό οι πληροφορίες παραμένουν ασφαλείς ακόμα και απέναντι στο λειτουργικό σύστημα ή το BIOS. Επομένως το SGX θα πρέπει να προστατεύει την εμπιστευτικότητα και την ακεραιότητα των enclaves. Το λογισμικό του συστήματος ή άλλες εφαρμογές δεν θα πρέπει να είναι σε θέση να λάβουν πληροφορίες που βρίσκονται εντός του enclave ή να διαχειριστούν τον κώδικα και τα δεδομένα του. Άρα, το enclave πρέπει να λειτουργεί ως μία ξεχωριστή οντότητα εντός του συστήματος. Τα enclaves αποθηκεύονται μέσα σε μία ειδική περιοχή μνήμης η οποία ονομάζεται Enclave Page Cache (EPC). Αυτή η περιοχή μνήμης κρυπτογραφείται με την χρήση ενός νέου chip το οποίο ονομάζεται memory encryption engine (MEE). Με αυτό τον τρόπο σε περίπτωση που κάποιος επιτιθέμενος κάνει μία φυσική επίθεση στο μηχάνημα διαβάζοντας τα δεδομένα μεταφέρονται μέσω του memory bus θα διαβάσει μόνο κρυπτογραφημένα δεδομένα καθώς όπως προαναφέρθηκε η αποκρυπτογράφηση γίνεται μόνο εντός του επεξεργαστή. Στο σχήμα 2.2 διακρίνεται η απομόνωση της μνήμης. Φυσικά αυτός ο νέος τύπος μνήμης χρειάζεται και κάποιο αντίστοιχο page cache map. Αυτό λοιπόν υπάρχει στην τεχνολογία SGX και ονομάζεται enclave page cache map (EPCM).Αυτή η δομή λοιπόν είναι υπεύθυνη για να αποθηκεύει την κατάσταση των σελίδων. Βρίσκεται εντός της προστατευμένης μνήμης και περιορίζει το μέγεθος του Enclave Page Cache σε 128MB, το οποίο είναι και το μέγεθος που ορίζει το BIOS.

Το Enclave Page Cache βρίσκεται εντός της μνήμης DRAM και πιο συγκεκριμένα εντός της περιοχής Processor Reserved Memory (PRM) το οποίο αποτελεί κομμάτι μνήμης το οποίο είναι δεσμευμένο για τις λειτουργίες του επεξεργαστή. Η δομή EPCM βρίσκεται εντός του επεξεργαστή. Επειδή τα δεδομένα αποθηκεύονται εκτός του επεξεργαστή πρέπει να υπάρχει ένας τρόπος ο οποίος θα εξασφαλίζει την ασφάλειά τους. Έτσι αφού κρυπτογραφηθούν με τη βοήθεια του chip Memory Encryption Engine αποστέλλονται στην μνήμη PRM. Αντίστροφα για την αποκρυπτογράφηση τα δεδομένα μεταφέρονται από την PRM στον επεξεργαστή και ακολούθως στην MEE όπου και πραγματοποιείται η διαδικασία της αποκρυπτογράφησης. Έτσι λοιπόν συμπεραίνουμε όταν το μοναδικό μέρος το οποίο τα δεδομένα βρίσκονται στην αρχική τους μορφή είναι εντός του επεξεργαστή. Επιπλέον, το SGX έχει τη δυνα-

τότητα να φέρει μία σελίδα από το Enclave Page Cache, να την τοποθετήσει σε μη προστατευμένη μνήμη και στη συνέχεια να την επαναφέρει. Ακόμη και ο επεξεργαστής όμως που είναι και η μοναδική οντότητα που έχει πρόσβαση στα δεδομένα και στον κώδικα ενός enclave θα πρέπει να βρίσκεται σε μία ειδική κατάσταση ώστε να συμβεί αυτό. Αυτή η κατάσταση του επεξεργαστή ονομάζεται κατάσταση enclave. Αυτή η κατάσταση ενεργοποιείται όταν ο κώδικας που βρίσκεται εντός του enclave επιθυμεί να αποκτήσει πρόσβαση εντός της περιοχής EPC που διαθέτει το enclave όπου ανήκει. Επομένως υπάρχουν τρεις απαιτήσεις ώστε να υπάρξει πρόσβαση στην προστατευμένη περιοχή μνήμης. Πρώτον, θα πρέπει επεξεργαστής να βρίσκεται σε κατάσταση enclave. Δεύτερον, η σελίδα που ζητείται από το EPC θα πρέπει να ανήκει στο ίδιο enclave το οποίο ζητάει και την πρόσβαση. Τρίτον, θα πρέπει σελίδα να βρίσκεται σε σωστή εικονική μνήμη. Συμπεραίνουμε λοιπόν ότι ο επεξεργαστής είναι υπεύθυνος για την ορθή λειτουργία του συστήματος η πρόσβαση στη προστατευμένη μνήμη ενός enclave πραγματοποιείται μόνο με συγκεκριμένους κανόνες. Τέλος, όλη η διαδικασία της κρυπτογράφησης και αποκρυπτογράφησης γίνεται μόνο εντός του επεξεργαστή και μόνο από ειδικό υλικό το οποίο φυσικά μπορεί να παρέχει υψηλότερες εγγυήσεις ασφαλείας από κάποιο λογισμικό.

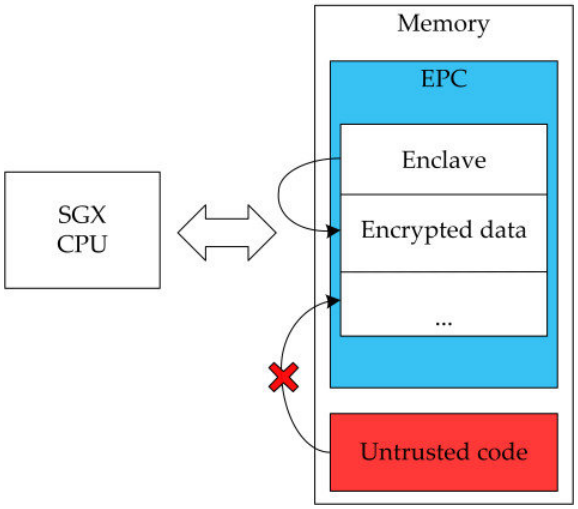

Figure 2.2: Απομόνωση της μνήμης.

2.8.7 Η δημιουργία ενός enclave

Η δημιουργία του enclave μπορεί να γίνει μόνο από το λειτουργικό σύστημα και μόνο με την εντολή ECREATE. Εφόσον η ασφάλεια του enclave βασίζεται στην κρυπτογράφηση θα πρέπει να υπάρχει και ο αντίστοιχος μηχανισμός που δημιουργεί τα κλειδιά της κρυπτογράφησης. Τα κλειδιά δημιουργούνται με την εντολή "EGETKEY" και εξαρτώνται από τους τρεις παρακάτω παράγοντες:

1. Κλειδί συσκευής: Πρόκειται για ένα κλειδί το οποίο δημιουργείται κατά την κατασκευή του επεξεργαστή έχει μέγεθος 128 bit και φυσικά βρίσκεται μόνο εντός του επεξεργαστή και είναι κρυφό.

2. Αριθμοί εκδόσεων ασφάλειας: Πρόκειται για αριθμούς που προσδιορίζουν την έκδοση του πυρήνα και έχουν μέγεθος 128 bit.

3. Owner Epoch: Είναι ένα κλειδί μεγέθους 128 bit το οποίο κατασκευάζει μια φορά το firmware του συστήματος και συσχετίζει το σύστημα με τον χρήστη.

2.8.8 Η δομή μια εφαρμογής σε Intel SGX

Για την κατασκευή μια εφαρμογής η οποία θα εκμεταλλεύεται την ύπαρξη της τεχνολογίας Intel SGX χρειάζεται να γίνει μια σημαντική παραδοχή. Αυτή είναι ότι η εφαρμογή χωρίζεται σε δύο τμήματα. Το ασφαλές τμήμα, το οποίο εκτελείται αποκλειστικά εντός του enclave, και το μη ασφαλές το οποίο εκτελείται εκτός.

Όλα τα τμήματα της εφαρμογής τα οποία διαχειρίζονται ευαίσθητα δεδομένα θα πρέπει να τοποθετούνται εντός του ασφαλούς τμήματος. Τα υπόλοιπα τμήματα θα πρέπει να τοποθετούνται στο μη ασφαλές τμήμα. Στο σχήμα 2.3 φαίνεται η δομή μιας εφαρμογής που χρησιμοποιεί Intel SGX. Μια πρώτη σημαντική παρατήρηση είναι πως θα πρέπει ο προγραμματιστής να κάνει μια προεργασία πριν κατασκευάσει την εφαρμογή του ώστε να αποφασίσει ποια κομμάτια επιβάλλεται να τοποθετηθούν εντός του ασφαλούς τμήματος και ποια εκτός. Μια δεύτερη σημαντική παρατήρηση είναι πως δεν είναι σωστή πρακτική ο προγραμματιστής να τοποθετεί όλα τα κομμάτια της εφαρμογής του εντός του ασφαλούς τμήματος. Αρχικά αυτό μπορεί να είναι αδύνατον καθώς το SGX διαθέτει περιορισμένο χώρο και δεύτερον αυτό θα προκαλούσε αναίτια επιβάρυνση στο σύστημα. Μεταξύ του ασφαλούς και του μη ασφαλούς τμήματος υπάρχει διεπαφή ώστε να μπορούν να αλληλοεπιδράσουν. Η

αλληλεπίδραση αυτή λοιπόν γίνεται με κλήσεις οι οποίες διαχωρίζονται σε e-calls και o-calls.

- E-calls: Είναι συναρτήσεις οι οποίες βρίσκονται εντός του enclave και καλούνται από τον κώδικα που βρίσκεται στο μη ασφαλές τμήμα.
- O-calls: Πρόκειται για συναρτήσεις οι οποίες βρίσκονται στο μη ασφαλές μέρος και καλούνται από το enclave.

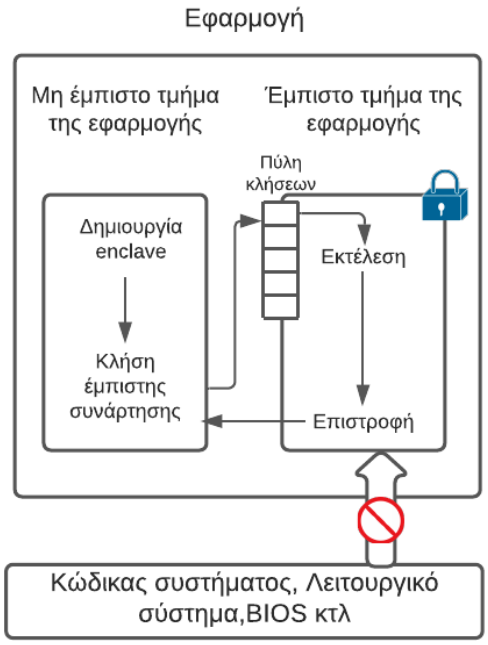

Figure 2.3: Διαχωρισμός ασφαλούς και ανασφαλούς τμήματος.

2.8.9 Δημιουργία μια εφαρμογής με Intel SGX

Η ίδια η εταιρεία Intel παρέχει στους προγραμματιστές ένα περιβάλλον κατασκευής λογισμικού (Software Development Kit ή εν συντομία SDK) [?] το οποίο μπορεί να χρησιμοποιηθεί για την κατασκευή μιας εφαρμογής με χρήση Intel SGX.

Σύμφωνα και με τους κατασκευαστές του Intel SGX [?] τα δεδομένα και ο κώδικας που φορτώνονται στο enclave πρέπει να έχουν το ελάχιστο δυνατό μέγεθος.

Μικρότερο μέγεθος οδηγεί σε μικρότερη επιφάνεια επίθεσης (attack surface). Γενικότερα είναι μέρος της φιλοσοφίας του SGX η ραγδαία μείωση του μεγέθους της επιφάνειας επίθεσης. Αυτό βέβαια το αναλαμβάνει κυρίως το SGX από μόνο του εφόσον θέτει εκτός της επιφάνειας επίθεσης τόσο το λειτουργικό σύστημα όσο και τον Virtual Machine Manager που είναι ο Hypervisor του συστήματος. Σκόπιμο είναι όμως και ο ίδιος ο προγραμματιστής να επενδύει χρόνο στην όσο πιο ασφαλή σχεδίαση της εφαρμογής ανεξάρτητα από τις εγγυήσεις ασφάλειας που παρέχει το σύστημα στο οποίο θα τρέχει η εφαρμογή. Επομένως, στην επιφάνεια της επίθεσης παραμένουν το κομμάτι του υλικού που αποτελεί το SGX καθώς και το ασφαλές τμήμα της εφαρμογής. Οπτικά οι παραπάνω παραδοχές παρουσιάζονται στο σχήμα 2.4.

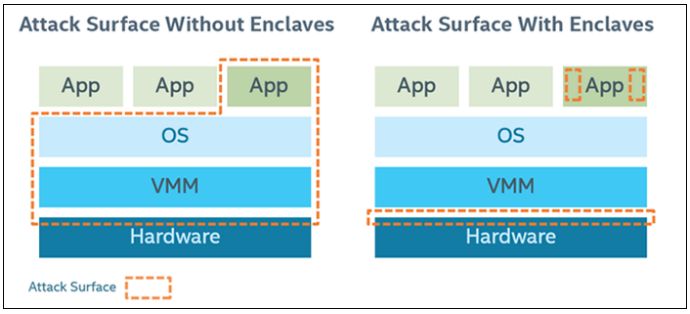

Figure 2.4: Επιφάνεια επίθεσης με enclave και χωρίς enclave.

2.8.10 Βασικές συναρτήσεις του Intel SGX

Μετά την επιτυχή εγκατάσταση του SDK που παρέχει η Intel ο προγραμματιστής μπορεί να ξεκινήσει την υλοποίηση της εφαρμογής του. Επειδή απαιτείται για λόγους ασφάλειας μια πολύ συγκεκριμένη δομή για την εφαρμογή παρέχεται ένα API. Αρχικά αυτό που πρέπει να γίνει πρώτα είναι η δημιουργία του enclave. Η συνάρτηση που παρέχεται για την παρακάτω λειτουργία είναι η sgx_create_enclave και παρουσιάζεται στο ακόλουθο σχήμα 2.5.

```
// Create an enclave
return_value = sgx_create_enclave(ENCLAVE_FILE, SGX_DEBUG_FLAG,
 &token, &token_updated, &eid, NULL);
```

Σχήμα 2.5: Η δημιουργία ενός enclave.

Η συγκεκριμένη συνάρτηση δέχεται τα παρακάτω ορίσματα:

- Enclave_file_name: Το όνομα ή το πλήρες μονοπάτι για το image του enclave.
- SGX_DEBUG_FLAG: Δυνατές τιμές είναι το 0 και το 1. Το 0 υποδηλώνει ότι το enclave δεν είναι σε κατάσταση εκσφαλμάτωσης. Η τιμή 1 δηλώνει ότι ο κώδικας και η μνήμη του enclave είναι προσβάσιμη από τον εκσφαλματωτή ή από εξωτερικό λογισμικό.
- Token: Αντικείμενο το οποίο απαιτείται για την δημιουργία του enclave και δεν μπορεί να είναι NULL. Ο δημιουργός θα πρέπει να το συγκρατεί ώστε σε μελλοντική χρήση να μπορεί να το χρησιμοποιήσει για την επαναδημιουργία του enclave.
- Updated: Προσδιορίζει αν το προηγούμενο token έχει ανανεωθεί. Ο αριθμός 0 δείχνει ότι δεν έχει υπάρξει ανανέωση ενώ ο αριθμός 1 ότι έχει γίνει.
- Eid: Αποτελεί το enclave id το οποίο είναι το αναγνωριστικό του enclave. Δεν επιτρέπεται να λάβει τιμή NULL.
- Misc_attr: Συγκρατεί πληροφορίες σχετικά με τα ορίσματα που έχει το enclave. Μπορεί να έχει την τιμή NULL αν αυτές οι πληροφορίες δεν απαιτούνται.

Μια ακόμη βασική συνάρτηση που χρησιμοποιείται στην παρούσα εργασία είναι η sgx_aes_ctr_encrypt η οποία πραγματοποιεί την κρυπτογράφηση. Πιο συγκεκριμένα υλοποιεί τον αλγόριθμο AES-CTR ο οποίος έχει παρουσιασθεί σε προηγούμενη ενότητα.

```
// Encrypt data with AES-CTR
sgx_status_t SGXAPI sgx_aes_ctr_encrypt(
    const sgx_aes_ctr_128bit_key_t *p_key,
    const uint8_t *p_src,
    const uint32_t src_len,
    uint8_t *p_ctr,
    const uint32_t ctr_inc_bits,
    uint8_t *p_dst
);
```

Σχήμα 2.6: Η συνάρτηση κρυπτογράφησης των δεδομένων.

Η συγκεκριμένη συνάρτηση δέχεται τα παρακάτω ορίσματα:

- p_key: Δείκτης στο κλειδί που χρησιμοποιείται για την κρυπτογράφηση. Το μέγεθος πρέπει να είναι υποχρεωτικά 128 bits.
- pc_src: Δείκτης στην ροή των δεδομένων τα οποία θα κρυπτογραφηθούν.
- src_len: Καθορίζει το μέγεθος των δεδομένων τα οποία θα κρυπτογραφηθούν.
- p_ctr: Δείκτης στο διάνυσμα αρχικοποίησης το οποίο θα χρησιμοποιηθεί για τους υπολογισμούς του αλγορίθμου AES-CTR.
- ctr_inc_bits: Προσδιορίζει τον αριθμό των bit του μετρητή που θα αρχικοποιηθεί και θα χρησιμοποιηθεί από τον αλγόριθμο κρυπτογράφησης.
- p_dst: Δείκτης στην δομή που θα αποθηκευτούν τα κρυπτογραφημένα δεδομένα.

Οι επιστρεφόμενες τιμές είναι αντίστοιχες με αυτές που επιστρέφονται από την συνάρτηση δημιουργίας ενός enclave. Σε περίπτωση επιτυχίας επιστρέφεται η τιμή SGX_SUCCESS. Υπάρχουν 3 δυνατές τιμές σε περίπτωση αποτυχίας. Η πρώτη είναι η SGX_ERROR_INVALID_PARAMETER η οποία επιστρέφεται σε περίπτωση που το κλειδί, η πηγή, ο προορισμός ή ο μετρητής έχουν την τιμή NULL. Η δεύτερη είναι η SGX_ERROR_OUT_OF_MEMORY η οποία δηλώνει ότι δεν υπάρχει ο απαιτούμενος χώρος για την ολοκλήρωση της λειτουργίας της κρυπτογράφησης. Τέλος, υπάρχει η τιμή SGX_ERROR_UNEXPECTED η οποία δηλώνει ότι υπάρχει κάποιο εσωτερικό λάθος στην βιβλιοθήκη που υλοποιεί την κρυπτογράφηση. Συμπερασματικά, η συνάρτηση sgx_aes_ctr_encrypt υλοποιεί τον αλγόριθμο κρυπτογράφησης AES-CTR και χρησιμοποιείται στην παρούσα εργασία με σκοπό την κρυπτογράφηση των δεδομένων εντός του enclave. Μια τελευταία σημαντική συνάρτηση που χρησιμοποιείται στην παρούσα εργασία είναι η συνάρτηση sgx_aes_ctr_decrypt η οποία υλοποιεί την διαδικασία της αποκρυπτογράφησης των δεδομένων και διακρίνεται στο σχήμα 2.7. Έχει την ίδια δομή και δέχεται τα ίδια ακριβώς ορίσματα με την συνάρτηση sgx_aes_ctr_encrypt. Επίσης, έχουν τις ίδιες δυνατές επιστρεφόμενες τιμές.

```
// Decrypt data with AES-CTR
sgx_status_t SGXAPI sgx_aes_ctr_decrypt(
    const sgx_aes_ctr_128bit_key_t *p_key,
```

```
    const uint8_t *p_src,
    const uint32_t src_len,
    uint8_t *p_ctr,
    const uint32_t ctr_inc_bits,
    uint8_t *p_dst
);
```

Σχήμα 2.7: Η συνάρτηση αποκρυπτογράφησης των δεδομένων.

2.8.11 Επιβεβαίωση ορθής λειτουργίας

Μια βασική λειτουργία ενός enclave είναι να επιβεβαιώνει την ορθή λειτουργία του τόσο σε enclaves τα οποία βρίσκονται στο ίδιο μηχάνημα όσο και σε εξωτερικές οντότητες του συστήματος. Αυτή η διαδικασία είναι ιδιαίτερα σημαντική καθώς αν δεν υπάρξει η απαραίτητη επιβεβαίωση τότε δεν μπορεί κανείς να είναι σίγουρος για την αξιόπιστη λειτουργία του enclave. Υπάρχουν δύο ειδών επιβεβαιώσεις. Η πρώτη είναι η τοπική επιβεβαίωση (local attestation) και η δεύτερη είναι η απομακρυσμένη επιβεβαίωση (remote attestation).

Ανάγκη για τοπική επιβεβαίωση μπορεί να υπάρξει όταν η εφαρμογή ενός προγραμματιστή απαιτεί την συνεργασία μεταξύ των enclaves. Σε αυτήν την περίπτωση λοιπόν απαιτείται η ύπαρξη ενός μηχανισμού ο οποίος θα επιβεβαιώνει την ταυτότητα και την αυθεντικότητα ενός enclave τοπικά στην υπόλοιπη πλατφόρμα. Το Intel SGX παρέχει έναν τέτοιο μηχανισμό με την βοήθεια του κατάλληλου υλικού. Αυτό που γίνεται λοιπόν είναι να ζητά το ίδιο το enclave την παραγωγή μιας αναφοράς (report) η οποία παρέχει κρυπτογραφικές αποδείξεις ότι το συγκεκριμένο enclave υπάρχει στην πλατφόρμα. Στην συνέχεια η αναφορά αυτή μπορεί να αποσταλεί σε ένα άλλο enclave της ίδιας πλατφόρμας. Κατά την επικοινωνία δύο enclaves θα πρέπει κα τα δύο να αποστείλουν αναφορά το ένα στο άλλο για επιβεβαίωση. Για την παραγωγή της αναφοράς χρησιμοποιείται ένα συμμετρικό κλειδί που μόνο το συγκεκριμένο enclave μπορεί να κατασκευάσει. Η αναφορά περιέχει τα εξής δεδομένα:

- Έλεγχος του κατακερματισμού (hash) του κώδικα καθώς και των δεδομένων του enclave

- Ένα hash του δημόσιου κλειδιού το οποίο δημιουργείται κατά την αρχικοποίηση του enclave
- Πληροφορίες ασφάλειας σχετικά με την κατάσταση του μηνύματος
- Υπογραφή των παραπάνω δεδομένων η οποία μπορεί να επαληθευτεί από την ίδια πλατφόρμα στην οποία παράχθηκε η αναφορά

Επιπλέον, υπάρχει η δυνατότητα της απομακρυσμένης επιβεβαίωσης (remote attestation). Σε αυτήν την διαδικασία η εφαρμογή η οποία χρησιμοποιεί το enclave θα χρειαστεί να του ζητήσει να παράξει μια αναφορά και στην συνέχεια να μεταβιβάσει αυτήν την αναφορά στην πλατφόρμα που φιλοξενεί το enclave. Ακολούθως, η πλατφόρμα παράγει ένα αποδεικτικό το οποίο επιβεβαιώνει τόσο το συγκεκριμένο enclave όσο και την κατάσταση της πλατφόρμας. Την συνέχεια το αποδεικτικό αυτό μπορεί να μεταφερθεί σε οντότητες εκτός της πλατφόρμας.

Η ίδια η Intel παρέχει μια υπηρεσία για τη επαλήθευση των υπογραφών η οποία ονομάζεται Intel Enhanced Privacy ID. Το αποδεικτικό το οποίο αποστέλλεται περιέχει τις παρακάτω πληροφορίες επιπλέον αυτών της αναφοράς:

- Το ID της συσκευής καθώς και τον αριθμό έκδοσης ασφάλειας του enclave (SVN)
- Πληροφορίες όπως για παράδειγμα τον τρόπο που λειτουργεί το enclave π.χ. debug mode

Chapter 3

Σχεδιαση

3.1 Σχεδίαση

Σε αυτό το κεφάλαιο θα παρουσιαστούν οι παραδοχές τις σχεδίασης, οι οντότητες του συστήματος και οι πιθανές επιθέσεις με τα αποτελέσματα τους. Στην συνέχεια θα παρουσιαστεί η σχεδίαση του συστήματος.

3.2 Οι παραδοχές της σχεδίασης

Αρχικά, πριν κάποιος μελετήσει με λεπτομέρεια την σχεδίαση του συστήματος θα πρέπει να γνωρίζει τις βασικές παραδοχές καθώς και ποιες οντότητες θεωρούνται έμπιστες και ποιες όχι. Οι παραδοχές θα πρέπει να είναι γνωστές από την αρχή ώστε κάθε χρήστης να γνωρίζει ποια είναι τα ασφαλή και ποια τα μη ασφαλή τμήματα της υπηρεσίας. Αρχικά, παρουσιάζουμε τις οντότητες που κάθε χρήστης θα πρέπει να θεωρεί έμπιστες:

1. Ο επεξεργαστής της τερματικής συσκευής: Οι επεξεργαστές οι οποίοι μπορούν να εκτελέσουν τις διαδικασίες κρυπτογράφησης και αποκρυπτογράφησης πρέπει υποχρεωτικά να υποστηρίζουν την τεχνολογία Intel SGX. Επομένως, θα πρέπει να υπάρχει εμπιστοσύνη στην εταιρεία Intel η οποία κατασκευάζει τους επεξεργαστές αυτού του είδους. Για παράδειγμα η εταιρεία ισχυρίζεται ότι δεν γνωρίζει το μυστικό κλειδί το οποίο έχει κάθε επεξεργαστής και το οποίο δημιουργείται και αποθηκεύεται, κατά την διαδικασία της κατασκευής, εντός του επεξεργαστή. Προφανώς, ο ιδιοκτήτης του επεξεργαστή δεν έχει κάποιο τρόπο να επιβεβαιώσει αυτόν τον ισχυρισμό.

2. Η κρυπτογραφική μέθοδος: Η κρυπτογράφηση των δεδομένων γίνεται με χρήση του αλγορίθμου AES-CTR τον οποίο υποστηρίζει το Intel SGX. Ο αλγόριθμος αυτός έχει αξιολογηθεί και επαληθευτεί από την επιστημονική κοινότητα επανειλημμένως. Επομένως, δεν υπάρχει λόγος να μην θεωρείται έμπιστος.

3. Το σύστημα αποθήκευσης του μηχανήματος: Τα δεδομένα που κρυπτογραφούνται αποθηκεύονται στην συσκευή. Επομένως, θα πρέπει ο χρήστης να εμπιστεύεται την συσκευή όσον αφορά την σωστή αποθήκευση των δεδομένων.

Εκτός των παραπάνω υποθέσεων ο χρήστης μπορεί να υποθέσει ότι ο πάροχος είναι αναξιόπιστος αλλά δεν επηρεάζει την συνολική ασφάλεια των δεδομένων εφόσον η κρυπτογράφηση των δεδομένων γίνεται εντός του SGX το οποίο φυσικά θεωρείται έμπιστο. Η παραδοχή αυτή πηγάζει από το γεγονός πως ο πάροχος λόγω του σχεδιασμού της τεχνολογίας SGX δεν έχει πρόσβαση στην κρυπτογραφική διαδικασία. Ο χρήστης δεν εμπιστεύεται λοιπόν τον πάροχο της υπηρεσίας νέφους στην οποία αποθηκεύονται τα κρυπτογραφημένα δεδομένα. Καθώς τα δεδομένα σε καμία στιγμή δεν βρίσκονται εκτεθειμένα σε κάποια οντότητα εκτός του χρήστη δεν

χρειάζεται κάποιος να εμπιστεύεται τον πάροχο ή οποιονδήποτε διαχειριστή της υπηρεσίας.

3.2.1 Το Μοντέλο Απειλών (Threat Model)

Το μοντέλο απειλών είναι μια διαδικασία καταγραφής πιθανών κινδύνων όπως δομικές ευπάθειες του συστήματος ή ευπάθειες του λογισμικού το οποίο χρησιμοποιείται κατά την διάρκεια της λειτουργίας του συστήματος. Θα πρέπει όλοι οι παράγοντες, που πιθανόν να απειλήσουν το σύστημα, να έχουν αναλυθεί πριν πραγματοποιηθεί η περιγραφή της σχεδίασης. Πρώτα, είναι απαραίτητο να παρουσιαστούν οι βασικοί όροι του μοντέλου απειλών. Οι 4 βασικοί όροι του μοντέλου απειλών είναι η Ευπάθεια, η Απειλή, η Επίθεση και η Εκμετάλλευση. Αρχικά η Ευπάθεια (Vulnerability) αφορά σημεία του συστήματος τα οποία είναι ευάλωτα και μια πιθανή επίθεση από κάποια κακόβουλη οντότητα μπορεί η ασφάλεια να τεθεί σε αμφισβήτηση. Πρόκειται για ατέλειες οι οποίες είτε προκύπτουν κατά την σχεδίαση ενός συστήματος είτε σε μεταγενέστερο χρόνο. Ο όρος Απειλή (Threat) αφορά ένα εξωτερικό κίνδυνο του συστήματος ο οποίος μπορεί να εξελιχθεί σε επίθεση και να εκμεταλλευθεί κάποια υπάρχουσα ευπάθεια. Λόγω της ύπαρξης κάποιας ευπάθειας δημιουργείται αντίστοιχα κάποια απειλή η οποία στο επόμενο στάδιο μπορεί να οδηγήσει σε επίθεση.Με τον όρο Επίθεση (Attack) εννοείται η προσπάθεια υλοποίησης μιας απειλής με σκοπό την εκμετάλλευση κάποιας ευπάθειας του συστήματος. Μια επίθεση έχει ως στόχο την διείσδυση εσωτερικά του συστήματος με σκοπό την απόκτηση του ελέγχου ή και των δεδομένων που διαθέτει το σύστημα. Μια επιτυχημένη επίθεση μπορεί να οδηγήσει στην Εκμετάλλευση. Ο όρος Εκμετάλλευση (Exploit) είναι η αξιοποίηση των αποτελεσμάτων μιας επιτυχούς επίθεσης. Τα αποτελέσματα όπως προαναφέρθηκαν μπορεί να είναι ο έλεγχος του συστήματος ή η πρόσβαση σε δεδομένα. Στην συνέχεια το σύστημα ή τα δεδομένα μπορούν να αξιοποιηθούν με κακόβουλο τρόπο από κάποια κακόβουλη εξωτερική οντότητα. Έχοντας δώσει τους παραπάνω ορισμούς μπορούμε πλέον να περιγράψουμε τις ευπάθειες της σχεδίασης που μπορούν να οδηγήσουν σε απειλές οι οποίες με την σειρά τους θα μετουσιωθούν σε επιθέσεις για την εκμετάλλευση του συστήματος ή των δεδομένων.

3.2.2 Ευπάθειες στην πλευρά του χρήστη

1. Κάθε χρήστης θα πρέπει να στέλνει το αρχείο στον διακομιστή ο οποίος θα το κρυπτογραφεί και θα το αποθηκεύει μέχρι αυτό να ζητηθεί ξανά. Επομένως ο χρήστης θα πρέπει να εξασφαλίσει την ασφάλεια του τοπικού του μηχανήματος. Το αρχείο αρχικά βρίσκεται στην αρχική του μορφή επομένως πιθανή παραβίαση της συσκευής θα εκθέσει το αρχείο στον επιτηθέμενο. Επομένως ο χρήστης είναι υπεύθυνος για την ασφάλεια των δεδομένων του μέχρι να χρησιμοποιήσει την υπηρεσία. Φυσικά αυτή η παραδοχή είναι λογικό να υπάρχει για οποιαδήποτε υπηρεσία καθώς δεν έχει πρόσβαση στα δεδομένα πριν χρησιμοποιηθεί. Μπορεί ο χρήστης μετά την αποστολή του αρχείου στον διακομιστή να διαγράψει το τοπικό αντίγραφο αφού πλέον το αρχείο βρίσκεται στον διακομιστή και μπορεί να γίνει προσβάσιμο ανά πάσα στιγμή. Αν επιλέξει να το κρατήσει και τοπικά τότε θα πρέπει να εξασφαλίσει μόνος του την ιδιωτικότητα των δεδομένων.

2. Θα πρέπει ο χρήστης να προστατεύει την τερματική του συσκευή και από επιθέσεις που έχουν σκοπό να πάρουν τον έλεγχο του μηχανήματος. Σε περίπτωση επιτυχίας μιας τέτοιας επίθεσης μπορεί ο επιτιθέμενος να προσποιηθεί ότι είναι ο ίδιος ο χρήστης. Αυτό φυσικά θα μπορούσε να έχει συνέπειες για την συνολική λειτουργία της υπηρεσίας.

3.2.3 Ευπάθειες στην πλευρά του διακομιστή

1. Στον διακομιστή φτάνουν συνεχόμενα μέσω TLS τα υποαρχεία του χρήστη τα οποία προωθούνται απευθείας στο SGX. Δεν υπάρχει δυνατότητα αποκρυπτογράφησης των δεδομένων καθώς η επικποινωνία είναι ουσιαστικά ανάμεσα στην συσκευή του χρήστη και το SGX. Στην συνέχεια το SGX κάνει αποκρυπτογράφηση με το κλειδί που έχει προκύψει από την διαδικασία του TLS και στην συνέχεια κρυπτογραφεί με το κλειδί που διαθέτει από το υλικό. Επομένως, σε καμία περίπτωση ο διακομιστής δεν έχει πρόσβαση στην αρχική μορφή των δεδομένων. Αφού, κρυπτογραφήσει το αρχείο που παρέλαβε το αποθηκεύει ώστε όταν του ζητηθεί εκ νέου να το αποκρυπτογραφήσει και να το εναναπροωθήσει στον χρήστη. Η πρώτη ευπάθεια λοιπόν που μπορεί να αποκτήσει κάποια εξωτερική οντότητα τον έλεγχο του μηχανήματος που πραγματοποιεί την κρυπτογράφηση και την αποθήκευση των δεδομένων. Σε

μια τέτοια περίπτωση ο πάροχος δεν μπορεί πλέον να προσφέρει την υπηρεσία ασφαλούς αποθήκευσης στους χρήστες αλλά δεν θα εκθέσει τα δεδομένα του χρήστη γεγονός που αποτελεί και την βασική εγγύηση της παρούσας υλοποίησης. Επομένως, απαιτείται από τον πάροχο να λάβει μέτρα προστασίας απέναντι σε επιθέσεις προς τα μηχανήματα του.

2. Εφόσον χρησιμοποιείται η τεχνολογία Intel SGX για την κρυπτογράφηση των αρχείων αυτομάτως οι ευπάθειες που έχει αυτή η τεχνολογία υιοθετούνται και από το ίδιο το σύστημα. Για παράδειγμα η τεχνολογία Intel SGX είναι ευάλωτη απέναντι σε επιθέσεις side channel.

3.2.4 Απειλές του συστήματος

1. Η απώλεια ελέγχου του συστήματος μπορεί να οδηγήσει σε πολύ αρνητικές συνέπειες όπως η διακοπή της λειτουργίας της υπηρεσίας. Αυτό μπορεί να γίνει με διάφορους τρόπους είτε απομακρυσμένα είτε με φυσική πρόσβαση στα μηχανήματα. Επομένως, θα πρέπει να ληφθούν μέτρα για την αποτροπή τέτοιων επιθέσεων. Θα πρέπει δηλαδή να υπάρχει αυστηρός έλεγχος πρόσβασης και για την απομακρυσμένη και για την φυσική πρόσβαση στα μηχανήματα.

2. Εφόσον το σύστημα δέχεται τα αρχεία των χρηστών θα πρέπει να προβλέπεται άμυνα απέναντι σε επιθέσεις κατανεμημένης άρνησης εξυπηρέτησης (Distributed Denial Of Service). Αυτό είναι ένα γνωστό πρόβλημα για πολλά χρόνια και υπάρχουν πλέον διάφορες πρακτικές λύσεις.

3.2.5 Εκμετάλλευση

Οι παραπάνω επιτυχείς επιθέσεις ή και άλλες μπορούν να οδηγήσουν στην απώλεια του ελέγχου του συστήματος ή στην απώλεια/γνωστοποίηση έμπιστων πληροφοριών. Οι συνέπειες των παραπάνω καταστάσεων είναι ανυπολόγιστες. Για παράδειγμα μπορεί να υπάρξει διακοπή μιας κρίσιμης υπηρεσίας προς τους χρήστες ή ακόμη και οι πληροφορίες να γίνονται αντικείμενο εκμετάλλευσης ή αγοραπωλησίας. Στην παρούσα υλοποίηση κάποια επιτυχημένη επίθεση θα έδινε πρόσβαση των αρχείων που κρυπτογραφούνται στον επιτιθέμενο. Επομένως, απαιτείται πέρα από σωστή σχεδίαση και χρήση μιας ασφαλούς τεχνολογίας να υπάρχει και συνεχής εξέλιξη της

καθώς και παρακολούθηση για τυχόν αντιμετώπιση κινδύνων και επιθέσεων που ήταν αδύνατον να προβλεφθούν σε προγενέστερο χρόνο.

3.3 Τα αρχεία που κρυπτογραφούνται

Σκοπός της παρούσας υπηρεσίας είναι η κρυπτογράφηση αρχείων με χρήση της τεχνολογίας Intel SGX. Αν και ο σχεδιασμός του SGX θέτει περιορισμό στο μέγεθος των δεδομένων που μπορούν να εισαχθούν σε κάποιο enclave στην παρούσα εργασία θα παρουσιάσουμε έναν μηχανισμό για την παράκαμψη αυτού του περιορισμού. Πιο συγκεκριμένα αν και η Intel δίνει σε κάθε enclave μνήμη μεγέθους 128MB στην πραγματικότητα τα ωφέλιμα είναι αρκετά λιγότερα καθώς αρκετά χρησιμοποιούνται για εσωτερικές διεργασίες του enclave καθώς και για αρχικοποιήσεις δομών δεδομένων. Επομένως, καθώς υπάρχει αυτός ο περιορισμός θα πρέπει το αρχείο να διαχωριστεί σε μικρότερα υποαρχεία τα οποία θα κρυπτογραφούνται ξεχωριστά το καθένα. Στην επόμενη ενότητα θα παρουσιαστεί αναλυτικά αυτή η διαδικασία.

3.4 Διάσπαση των αρχείων

Όπως προαναφέρθηκε είναι υποχρεωτικό λόγω του περιορισμού μεγέθους της μνήμης που εισάγει το SGX θα πρέπει το αρχικό αρχείο να διασπαστεί σε μικρότερα υποαρχεία τα οποία ονομάζονται chunks. Το μέγεθος των υποαρχείων είναι μεταβλητό αλλά όλα τα αρχεία έχουν το μέγεθος που καθορίστηκε, εκτός από το τελευταίο υποαρχείο, το οποίο δεν θα είναι πλήρες στην περίπτωση που το μέγεθος του αρχικού αρχείου δεν διαιρείται ακέραια με το μέγεθος που θέσαμε σαν αρχικό περιορισμό. Δεν γίνεται κάποια επεξεργασία στο αρχικό αρχείο ούτε και στα υποαρχεία. Τα δεδομένα δηλαδή μένουν ακέραια. Επομένως, η παραπάνω διαδικασία δεν δημιουργεί κάποιο κενό ασφάλειας. Στην συνέχεια κάθε υποαρχείο κρυπτογραφείται ξεχωριστά από τα υπόλοιπα εντός του enclave με το ίδιο κλειδί. Την διαδικασία της διάσπασης του αρχείου την αναλαμβάνει η τερματική συσκευή του χρήστη. Αυτό είναι απαραίτητο να συμβεί στην πλευρά του χρήστη ώστε τα υποαρχεία να φτάνουν απευθείας στο SGX που διαθέτει ο διακομιστής έχοντας κρυπτογραφηθεί μέσω του TLS. Αν όλο το αρχείο έφτανε στον διακομιστή, ώστε

εκείνος να κάνει την διάσπαση, τότε τα δύο άκρα της επικοινωνίας, με χρήση TLS, θα ήταν η συσκευή του χρήστη με τον διακομιστή και όχι συγκεκριμένα με το SGX που διαθέτει ο διακομιστής. Η διαφορά εκ πρώτης όψεως ίσως φαίνεται μικρή αλλά στην πράξη είναι ιδιαίτερα ουσιώδης καθώς αν η ροή των δεδομένων δεν φτάνει απευθείας στο SGX τότε ο διακομιστής θα είναι σε θέση να δει αποκρυπτογραφημένα τα δεδομένα του χρήστη. Συμπερασματικά, η διάσπαση του αρχείου γίνεται στην συσκευή του χρήστη γεγονός που στην συνέχεια εξασφαλίζει ότι ο διακομιστής δεν θα έχει σε καμία στιγμή πρόσβαση στα δεδομένα του χρήστη.

3.5 Κωδικοποίηση των αρχείων

Δεν θεωρείται δεδομένο ότι τα αρχεία τα οποία θα πρέπει να κρυπτογραφηθούν θα έχουν και την κατάλληλη μορφή. Προαπαιτούμενο για να πραγματοποιηθεί σωστά η κρυπτογράφηση του αρχείου είναι το αρχείο να έχει κωδικοποίηση base64. Επομένως θα πρέπει να ελεγχθεί αν το δυαδικό αρχείο το οποίο πρόκειται να κρυπτογραφηθεί έχει και την κατάλληλη κωδικοποίηση. Επομένως αφού ολοκληρωθεί η διαδικασία της διάσπασης του αρχείου σε υποαρχεία θα πρέπει να ελεγχθεί αν τα υποαρχεία έχουν κωδικοποίηση base64 και σε περίπτωση που δεν έχουν τότε θα πρέπει να γίνει η απαραίτητη μετατροπή. Επομένως, γίνεται έλεγχος στο πρώτο υποαρχείο. Αν αυτό έχει την σωστή κωδικοποίηση τότε δεν υπάρχει λόγος να πραγματοποιηθεί έλεγχος και στα υπόλοιπα. Αν δεν την έχει τότε το σύστημα αναλαμβάνει να το κωδικοποιήσει. Επομένως, πραγματοποιείται έλεγχος για την μορφή του αρχείου και αν αυτό δεν την έχει τότε πραγματοποιείται η απαραίτητη διαδικασία Συμπερασματικά, μια λάθος μορφή δεν αποτελεί πρόβλημα για την υλοποίηση μας καθώς έχει ληφθεί υπόψιν και έχει διορθωθεί.

3.6 Κρυπτογράφηση κάθε υποαρχείου εντός του Intel SGX

Τα υποαρχεία φτάνουν σειριακά στο SGX. Επομένως είναι πλέον δυνατή η κρυπτογράφηση κάθε υποαρχείου ξεχωριστά. Επειδή προφανώς τα υποαρχεία έχουν πολύ μικρότερο μέγεθος από το αρχικό αρχείο είναι πλέον δυνατή η εισαγωγή τους σε κάποιο enclave. Επομένως, παρακάμφθηκε ο αρχικός περιορισμός του μεγέθους

μνήμης του SGX. Μετά την κρυπτογράφηση θα γίνει σύνθεση των υποαρχείων ώστε να παραχθεί ένα νέο ενιαίο αρχείο το οποίο είναι η κρυπτογραφημένη μορφή του αρχικού αρχείου πριν διασπαστεί. Μετά την κρυπτογράφηση ενός υποαρχείου εντός του SGX προκύπτει ένα κρυπτογραφημένο υποαρχείο. Όλα τα κρυπτογραφημένα υποαρχεία αποθηκεύονται τοπικά μέχρι να γίνει η σύνθεση τους. Η συγκεκριμένη διαδικασία είναι και το πιο ουσιώδες κομμάτι της παρούσας εργασίας. Η κρυπτογράφηση με χρήση ειδικού υλικού, το οποίο στην προκειμένη περίπτωση είναι το Intel SGX, είναι ένα αποφασιστικό βήμα προς την εξασφάλιση της εμπιστευτικότητας των δεδομένων. Στο σχήμα 3.1 παρουσιάζεται η διαδικασία κρυπτογράφησης των υποαρχείων εντός του SGX.

Μετά την κρυπτογράφηση τα δεδομένα έχουν κρυπτογραφηθεί πλέον με το μοναδικό κλειδί που βρίσκεται εντός του επεξεργαστή. Συνεπώς, ένας επιτιθέμενος ο οποίος με κάποιον τρόπο κατορθώσει να υποκλέψει τα δεδομένα που κατέχει ο πάροχος θα έρθει αντιμέτωπος με μια κρυπτογράφηση που δεν είναι κοινότυπη. Γίνεται λοιπόν κατανοητό το μέγεθος της δυσκολίας της κρυπτανάλυσης των δεδομένων γεγονός που αποτελεί το αποτέλεσμα και την προσφορά της παρούσας υλοποίησης στο πάντα ανοιχτό πρόβλημα της ασφάλειας και της ιδιωτικότητας των δεδομένων.

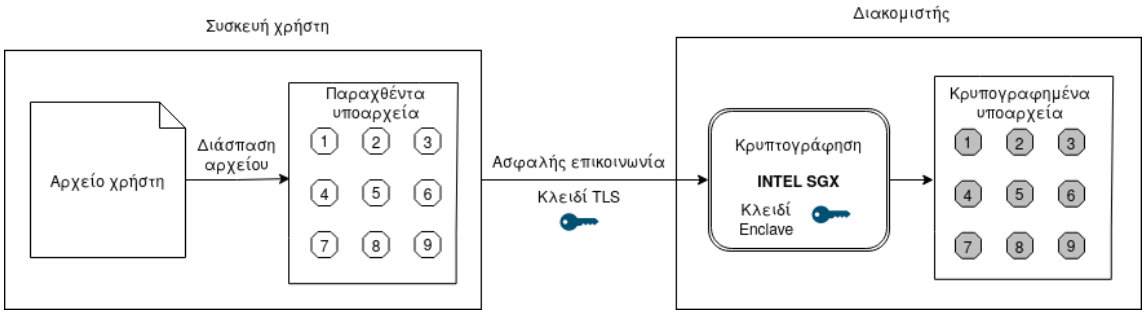

Figure 3.1: Η διάσπαση και η κρυπτογράφηση των υποαρχείων.

3.7 Αποκρυπτογράφηση κάθε υποαρχείου εντός του Intel SGX

Τα κρυπτογραφημένα υποαρχεία αποθηκεύονται τοπικά σε έναν αντίστοιχο φάκελο για όλα τα υποαρχεία του χρήστη. Όταν κάποιος χρήστης ζητήσει πρόσβαση

στο αρχικό αρχείο, το οποίο είχε στείλει, θα πρέπει πρώτα να γίνει η αποκρυπτογράφηση των κρυπτογραφημένων υποαρχείων και τελικά η σύνθεση εκ νέου του αρχικού αρχείου. Επομένως το πρώτο βήμα είναι ξεχωριστά κάθε κρυπτογραφημένο υποαρχείο να αποκρυπτογραφηθεί εντός του SGX. Στην συνέχεια, σειριακά όλα τα υποαρχεία που έχουν αποκρυπτογραφηθεί θα πρέπει να αποσταλούν με χρήση TSL στον χρήστη. Επομένως ακολουθείται η αντίστροφη διαδικασία από αυτήν που έγινε προηγουμένως για την κρυπτογράφηση. Στην συνέχεια στην πλευρά του χρήστη θα γίνει η σύνθεση των υποαρχείων και το τελικό αποτέλεσμα είναι το αρχικό αρχείο που είχε παραδώσει ο χρήστης στον διακομιστή. Η παραπάνω διαδικασία δεν θέτει τα δεδομένα σε κάποιον επιπλέον κίνδυνο καθώς η αποκρυπτογράφηση γίνεται εντός του enclave το οποίο δημιουργεί το Intel SGX για αυτόν τον σκοπό. Στο σχήμα 3.2 παρουσιάζεται η διαδικασία της αποκρυπτογράφησης και την σύνθεσης των κρυπτογραφημένων υποαρχείων σε ένα ενιαίο αρχείο.

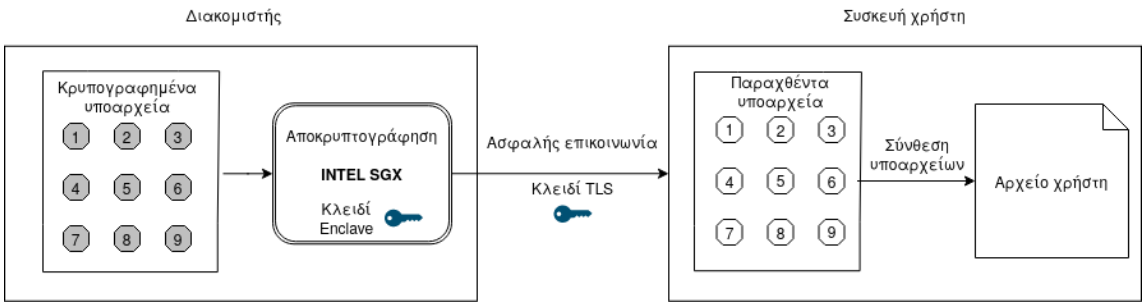

Figure 3.2: Η αποκρυπτογράφηση και η σύνθεση των υποαρχείων.

3.8 Σχηματική αναπαράσταση του συστήματος

Στο σχήμα 3.2 βλέπουμε ποια δεδομένα έχει κάθε οντότητα της υπηρεσίας. Ο χρήστης που βρίσκεται στα αριστερά διαθέτει το αρχείο ενώ η υπηρεσία διαθέτει τα κρυπτογραφημένα υποαρχεία που προκύπτουν από το αρχείο που λαμβάνει. Επιπλέον, γίνεται αντιληπτό ότι μετά την κρυπτογράφηση τα δεδομένα δεν βρίσκονται στην αρχική τους μορφή της οποία θα μπορούσε να εκμεταλλευθεί κάποιος επιτιθέμενος. Για να βρεθούν τα δεδομένα στην αρχική μορφή θα πρέπει αυτά να αποκρυπτογραφηθούν τοπικά στο μηχάνημα που διαθέτει την τεχνολογία SGX. Η

επικοινωνία μεταξύ των οντοτήτων πρέπει να είναι αμφίδρομη καθώς οποιαδήποτε στιγμή κάθε χρήστης έχει την δυνατότητα να ζητήσει το αρχείο ή τα αρχεία που έχει αποστείλει στην υπηρεσία.

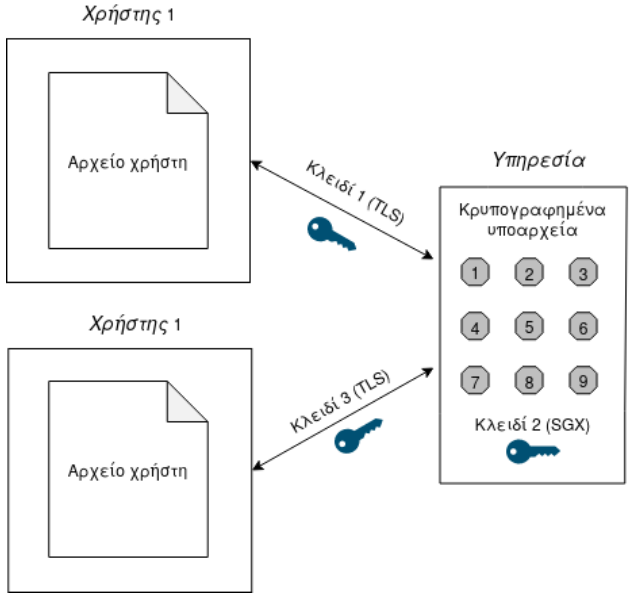

Figure 3.3: Σχηματική αναπαράσταση της υπηρεσίας.

3.9 Ομομορφική κρυπτογράφηση αρχείων

Όπως έχει αναφερθεί και πιο πριν η ομομορφική κρυπτογράφηση δεδομένων αποτελεί έναν σύγχρονο τρόπο κρυπτογράφησης που προσφέρει ορισμένες νέες ιδιότητες όπως για παράδειγμα η επεξεργασία των δεδομένων σε κρυπτογραφημένη μορφή. Εκτός λοιπόν από την κρυπτογράφηση με χρήση υλικού και συγκεκριμένα με χρήση της τεχνολογίας Intel SGX πραγματοποιήθηκε και κρυπτογράφηση των δεδομένων με χρήση πλήρους ομομορφικής κρυπτογράφησης. Σκοπός είναι η σύγκριση των κρυπτογραφικών μεθόδων ώστε να μπορέσει να γίνει μια εκτίμηση όσον αφορά την απόδοση τους. Σε αυτήν την περίπτωση δεν είναι απαραίτητη η διάσπαση του αρχείου. Η επικοινωνία μεταξύ της συσκευής του χρήστη και του παρόχου γίνεται και πάλι με TLS γεγονός που εξασφαλίζει πως κατά την επικοινωνία χρήστη και παρόχου τα δεδομένα είναι σε κρυπτογραφημένη μορφή. Όμως, μια σημαντική παραδοχή που πρέπει να γίνει είναι πως ο διακομιστής πριν πραγματοποιήσει την ομομορφική

κρυπτογράφηση μπορεί να αποκτήσει πρόσβαση στα δεδομένα του χρήστη. Στην περίπτωση της κρυπτογράφησης με χρήση SGX δεν υπήρχε αυτή η δυνατότητα καθώς ο διακομιστής δεν έχει πρόσβαση στις εσωτερικές διαδικασίες του SGX.

Μετά την παραλαβή του αρχείου γίνεται παραγωγή μυστικού ασύμμετρου κλειδιού το οποίο μπορεί να έχει μεταβλητό μέγεθος. Η παραγωγή του μυστικού κλειδιού γίνεται τοπικά στο μηχάνημα που θα πραγματοποιηθεί και η διαδικασία της κρυπτογράφησης. Μετά την παραγωγή του κλειδιού μπορεί πλέον να πραγματοποιηθεί η ομομορφική κρυπτογράφηση. Στην συνέχεια μπορεί σύμφωνα και με τις ιδιότητες τις ομομορφικής κρυπτογράφησης να γίνει συνδυασμός των κρυπτογραφημένων δεδομένων με άλλα αντίστοιχα κρυπτογραφημένα δεδομένα που έχουν κρυπτογραφηθεί με το ίδιο κλειδί. Έτσι λοιπόν μπορεί κάποιος να πραγματοποιήσει πράξεις με κρυπτογραφημένα δεδομένα χωρίς να έχει πρόσβαση στην αρχική μορφή των δεδομένων. Φυσικά, η αποκρυπτογράφηση γίνεται με το ασύμμετρο κλειδί. Στο σχήμα 3.4 παρουσιάζεται σχηματικά η διαδικασία της ομομορφικής κρυπτογράφησης.

Ένα σενάριο το οποίο θα εκμεταλλεύεται στην πράξη την βασική ιδιότητα της ομομορφικής κρυπτογράφησης, η οποία είναι οι πράξεις μεταξύ των κρυπτογραφημένων δεδομένων, είναι να αποστέλλει ένας χρήστης τα κρυπτογραφημένα δεδομένα σε έναν διακομιστή ο οποίος θα εκτελεί εκείνος τις απαραίτητες πράξεις. Με την προϋπόθεση λοιπόν ότι τα δεδομένα κρυπτογραφούνται με κοινό κλειδί μπορεί ο διακομιστής να παρέχει υπολογιστική ισχύ.

Επομένως η ομομορφική κρυπτογράφηση μπορεί να χρησιμοποιηθεί ως ένας ακόμα τρόπος για την ασφαλή αποθήκευση αλλά και επεξεργασία δεδομένων στην υπηρεσία και γενικότερα στο νέφος.

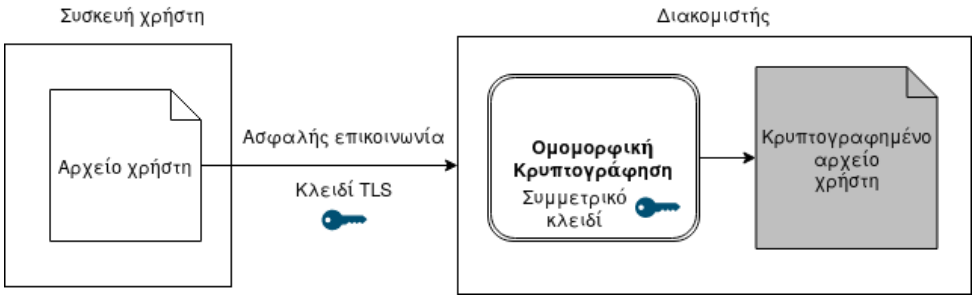

Figure 3.4: Η διαδικασία της ομομορφικής κρυπτογράφησης.

3.10 Συμμετρική κρυπτογράφηση με αξιοποίηση του υλικού

Εκτός των παραπάνω δύο υλοποιήσεων πραγματοποιήθηκε και υλοποίηση συμμετρικής κρυπτογράφησης με αξιοποίηση υλικού. Πιο συγκεκριμένα με κατάλληλη αξιοποίηση εντολών μηχανής που υποστηρίζει ο επεξεργαστής του συστήματος υλοποιήθηκε κρυπτογράφηση με αντίστοιχες εγγυήσεις ασφάλειας που εκτός από την ιδιωτικότητα επικεντρώνεται και στην απόδοση του αλγορίθμου. Επομένως κρίθηκε χρήσιμο να πραγματοποιηθεί σύγκριση και της συγκεκριμένης κρυπτογραφικής μεθόδου με τις προηγούμενες δύο υλοποιήσεις. Στο σχήμα 3.5 που ακολουθεί παρουσιάζονται οι κλήσεις συστήματος που βοηθούν στην βελτίωση της απόδοσης της κρυπτογράφησης και αποκρυπτογράφησης δεδομένων.

Εντολές
AES-New Instructions

AESENC	Εκτέλεση ενδιάμεσου κύκλου κρυπτογράφησης
AESENCLAST	Εκτέλεση τελευταίου κύκλου κρυπτογράφησης
AESDEC	Εκτέλεση ενδιάμεσου κύκλου αποκρυπτογράφησης
AESDECLAST	Εκτέλεση τελευταίου κύκλου αποκρυπτογράφησης
AESKEYGENASSIST	Υποβοήθηση στην παραγωγή του κλειδιού κρυπτογράφησης
AESIM	Υποβοήθηση στην αναστροφή στηλών (Inverse Mix Columns)

Figure 3.5: Οι εντολές AES-NI.

Chapter 4

Υλοποιηση

4.1 Υλοποίηση

Σε αυτό το κεφάλαιο παρουσιάζουμε την υλοποίηση των μεθόδων που περιγράφηκαν στο προηγούμενο κεφάλαιο της σχεδίασης. Θα αναλυθεί ο τρόπος με τον οποίο οι μέθοδοι διαχειρίζονται διαφορετικά μεγέθη αρχείων με σκοπό να ξεπεραστούν ορισμένοι περιορισμοί που εισάγονται από την αρχιτεκτονική του υλικού. Επιπλέον θα δείξουμε τις εντολές μηχανής που χρησιμοποιήθηκαν ώστε να επιτευχθεί η καλύτερη δυνατή απόδοση καθώς και άλλα μέρη του κώδικα τα οποία υλοποιούν βασικές λειτουργίες και παρουσιάζουν ενδιαφέρον. Πριν την παρουσίαση των παραπάνω θα γίνει μια αναφορά στην αποθήκευση των αρχείων στο μηχάνημα που χρησιμοποιήθηκε για την υλοποίηση.

4.2 Η αποθήκευση των αρχείων

Τα αρχεία αφού κρυπτογραφηθούν με οποιαδήποτε από τους τρόπους που περιγράφηκαν στην σχεδίαση αποθηκεύονται στο σύστημα αρχείων του συστήματος στην κρυπτογραφημένη τους μορφή. Επομένως ακόμη και αν κάποιος αποκτήσει πρόσβαση στο σύστημα αρχείων του συστήματος θα μπορέσει να αποκτήσει τα αρχεία τα οποία έχουν περάσει μια διαδικασία ισχυρής κρυπτογράφησης. Μάλιστα, ένας επιτιθέμενος δεν θα είναι σε θέση να προσδιορίσει την μέθοδο κρυπτογράφησης από την οποία πέρασε το αρχείο ώστε να χρησιμοποιήσει αυτήν την γνώση για την εκμετάλλευση κάποιας ευπάθειας των συγκεκριμένων υλοποιήσεων. Συμπερασματικά κατά την διάρκεια που τα αρχεία είναι αποθηκευμένα στο μηχάνημα δεν βρίσκονται στην αρχική αλλά στην κρυπτογραφημένη τους μορφή.

4.3 Ο περιορισμός της αρχιτεκτονικής του SGX

Όπως αναφέρθηκε και στο κεφάλαιο 3 η παρούσα αρχιτεκτονική της τεχνολογίας Intel SGX εισάγει έναν βασικό περιορισμό ο οποίος ήταν υποχρεωτικό να ληφθεί υπόψιν καθώς και να σχεδιαστεί τρόπος για την αποδοτική του παράκαμψη. Πιο συγκεκριμένα, ο περιορισμός αφορά στο μέγεθος της μνήμης του Intel SGX το οποίο αν και θεωρητικά είναι 128MB στην πράξη η μνήμη που μπορεί να χρησιμοποιηθεί και δεν είναι δεσμευμένη είναι λίγο πάνω από 90MB. Τα υπόλοιπα χρησιμοποιούνται από το Intel SGX για εσωτερικές διεργασίες. Το Intel SGX διαχειρίζεται από μόνο του αρχεία με μέγεθος ίσο με την μνήμη που έχει καθοριστεί από το BIOS δηλαδή αρχεία έως 128MB. Επομένως αν κάποιος επιθυμεί να χρησιμοποιήσει την παρούσα τεχνολογία για ένα αρχείο 200MB θεωρητικά δεν θα μπορέσει να το κάνει. Στην παρούσα εργασία ωστόσο παρουσιάσαμε τον τρόπο με τον οποίο μπορεί να ξεπεραστεί αυτός ο περιορισμός με την διάσπαση του αρχείου σε υποαρχεία τα οποία κρυπτογραφούνται ξεχωριστά. Όπως αναφέραμε και πριν η διάσπαση των αρχείων γίνεται στην συσκευή του χρήστη. Για αρχεία τα οποία έχουν μικρότερο μέγεθος από τον περιορισμό που έχει τεθεί δεν υπάρχει κάποιο πρόβλημα επομένως δεν απαιτείται η διάσπαση του αρχείου.

4.3.1 Ο έλεγχος της μορφής του αρχείου

Αυτόματα θα κληθεί η υπορουτίνα η οποία θα ελέγξει αν το αρχείο έχει σωστή κωδικοποίηση. Η υλοποίηση της διαδικασίας που θα περιγραφεί έχει υλοποιηθεί σε γλώσσα Python. Σε περίπτωση που το αρχείο δεν έχει την απαραίτητη κωδικοποίηση τότε το σύστημα αναλαμβάνει να το φέρει στην κατάλληλη μορφή. Πιο συγκεκριμένα απαιτείται κωδικοποίηση τύπου base64 η οποία αναπαριστά δεδομένα σε δυαδική μορφή. Φυσικά μια τέτοια επεξεργασία δεν θα αλλοιώσει το περιεχόμενο των δεδομένων. Σε περίπτωση που το αρχείο έχει διασπαστεί τότε θα πρέπει να κωδικοποιηθούν ανεξάρτητα μεταξύ τους όλα τα υποαρχεία που έχουν δημιουργηθεί. Σε περίπτωση που το αρχείο δεν διασπάστηκε τότε θα κωδικοποιηθεί ολόκληρο το αρχείο. Εν κατακλείδι, τα δεδομένα μετατρέπονται σε μια μορφή που είναι διαχειρίσιμη στο επόμενο στάδιο το οποίο είναι αυτό της κρυπτογράφησης.

Επομένως έχει πραγματοποιηθεί όλη η απαραίτητη προεργασία ώστε να χρησιμοποιηθεί το Intel SGX. Στο σχήμα 4.1 φαίνεται η κλήση της συνάρτησης Python η οποία πραγματοποιήθηκε ώστε να γίνει η απαραίτητη κωδικοποίηση.

`base64.b64encode(file_to_encode.read())`	Η συνάρτηση Python που λαμβάνει ως όρισμα ένα αρχείο και το κωδικοποιεί

Figure 4.1: Κλήση συνάρτησης Python για την κωδικοποίηση αρχείου.

4.4 Η κρυπτογράφηση κάθε αρχείου

Μετά την προεργασία που περιγράφηκε μπορεί να εκκινήσει η πραγματοποίηση της κρυπτογράφησης με χρήση του Intel SGX. Για την χρήση του Intel SGX αξιοποιήθηκε το αντίστοιχο SDK το οποίο παρέχει η ίδια η εταιρεία Intel και επιτρέπει την ανάπτυξη προγραμμάτων σε γλώσσα C++.

Η παραπάνω διαδικασία της διάσπασης μπορεί να οδηγήσει σε δύο περιπτώσεις. Πρώτα, σε περίπτωση που το αρχείο δεν διασπάστηκε. Τότε υπάρχει ένα μοναδικό αρχείο το οποίο κρυπτογραφηθεί. Δεύτερον, αν διασπάστηκε τότε υπάρχουν πολλαπλά αρχεία τα οποία θα πρέπει να κρυπτογραφηθούν. Η διαδικασία όμως

παραμένει η ίδια και στις δύο περιπτώσεις και το μόνο που αλλάζει είναι ο αριθμός των αρχείων. Αρχικά ας δούμε συνολικά στο σχήμα 4.2 το τμήμα του κώδικα που κρυπτογραφεί κάθε υποαρχείο και στην συνέχεια θα εξετάσουμε ξεχωριστά τα κυριότερα σημεία.

```
// Iterate through all chunkfiles
for(string name_of_original_chunkfile : vector_of_chunkfiles) {

    ifstream ifd(directory_to_check + name_of_original_chunkfile,
    ios::binary | ios::ate);
    size = ifd.tellg();
    ifd.seekg(0, ios::beg);
    buffer.resize(size); // << resize not reserve
    ifd.read(buffer.data(), size);
    message = buffer.data();

    // The encrypted message will contain the MAC, the IV,
    // and the encrypted message itself.
    size_t length_of_encrypted_message = (SGX_AESGCM_MAC_SIZE
    + SGX_AESGCM_IV_SIZE + strlen(message));
    char *encrypted_message = (char *)
    malloc((length_of_encrypted_message+1)*sizeof(char));

    // Encrypt each chunk and save it to the corresponding chunk file
    ret = encryptMessage(eid, message, strlen(message),
    encrypted_message, length_of_encrypted_message);

    // Write the encrypted data to a new chunkfile
    ofstream encrypted_chunkfile (name_of_new_encrypted_chunfile
    , ios::out | ios::binary);
    encrypted_chunkfile.write (encrypted_message);
    encrypted_chunkfile.close();
```

```
}
```

Σχήμα 4.2: Η κρυπτογράφηση κάθε υποαρχείου.

4.4.1 Η αρχικοποίηση του enclave

Η πρώτη ενέργεια η οποία πρέπει να πραγματοποιηθεί είναι η αρχικοποίηση του enclave η οποία παρουσιάζεται στο σχήμα 4.3. Αυτό φυσικά γίνεται με μοναδικό και ασφαλή τρόπο με χρήση βιβλιοθήκης η οποία παρέχεται απευθείας από την Intel. Αρχικά όπως έχει αναφερθεί και στο κεφάλαιο 2 η δημιουργία ενός enclave γίνεται με την μικροεντολή επεξεργαστή EADD. Σε ένα πραγματικό πρόγραμμα δεν καλείται απευθείας η συγκεκριμένη μικροεντολή αλλά η κλήση της πραγματοποιείται με την χρήση της συνάρτησης sgx_create_enclave. Η συγκεκριμένη συνάρτηση αρχικοποιεί το enclave και σε περίπτωση οποιουδήποτε λάθους διακόπτει την λειτουργία του προγράμματος και ενημερώνει τον προγραμματιστή για την αιτία της διακοπής.

```
// Initialize necessary structures for enclave creation
sgx_enclave_id_t eid;
sgx_status_t ret;
sgx_launch_token_t token = { 0 };
int token_updated = 0;

// Create an enclave
ret = sgx_create_enclave(ENCLAVE_FILE, SGX_DEBUG_FLAG, &token,
&token_updated, &eid, NULL);

 //check if creation was succesful
if (ret != SGX_SUCCESS)
{
    printf("sgx_create_enclave failed: %#x\n", ret);
    return 1;
}
```

Σχήμα 4.3: Κλήση συνάρτησης C++ για την δημιουργία ενός enclave.

4.4.2 Η κρυπτογράφηση με χρήση Intel SGX

Μετά την σωστή αρχικοποίηση του enclave πραγματοποιείται η κρυπτογράφηση των αρχείων. Σειριακά λοιπόν γίνεται η ανάγνωση των δεδομένων κάθε αρχείου και στην συνέχεια τα δεδομένα αυτά κρυπτογραφούνται με χρήση της τεχνολογίας Intel SGX. Για την πραγματοποίηση της κρυπτογράφησης χρησιμοποιήθηκε η συνάρτηση encryptMessage που παρουσιάζεται στο σχήμα 4.4 και η οποία λαμβάνει ως όρισμα το αναγνωριστικό κείμενο, τα αρχικά δεδομένα, το μέγεθος των αρχικών δεδομένων, έναν δείκτη στην δομή που θα αποθηκευτούν τα κρυπτογραφημένα δεδομένα καθώς και το μέγεθος της δομής που θα χρησιμοποιηθεί. Μετά την επιτυχή κλήση της παραπάνω συνάρτησης έχουμε πλέον τα δεδομένα σε κρυπτογραφημένη μορφή. Στην συνέχεια τα κρυπτογραφημένα δεδομένα αποθηκεύονται σε νέα αρχεία στο σύστημα αρχείων μέχρι να ζητηθεί η αποκρυπτογράφηση τους. Τέλος, διαγράφονται τα αρχεία που παρέλαβε το SGX μέσω του TLS. Επομένως πλέον στο σύστημα υπάρχουν τα δεδομένα στην ασφαλέστερη δυνατή μορφή εφόσον έχει γίνει η κρυπτογράφηση τους με την βοήθεια υλικού.

```
// The encrypted message will contain the MAC, the IV,
// and the encrypted message itself.
size_t length_of_encrypted_message = (SGX_AESGCM_MAC_SIZE
+ SGX_AESGCM_IV_SIZE + strlen(message));
char *encrypted_message = (char *)
malloc((length_of_encrypted_message+1)*sizeof(char));

// Encrypt each chunk and save it to the corresponding chunk file
ret = encryptMessage(eid, message, strlen(message),
encrypted_message, length_of_encrypted_message);
```

Σχήμα 4.4: Κλήση συνάρτησης C++ για την κρυπτογράφηση αρχείου.

4.5 Η αποκρυπτογράφηση κάθε αρχείου

Ανά πάσα στιγμή μπορεί να ζητηθεί η αποκρυπτογράφηση του αρχικού αρχείου που είχε δώσει κάποιος στο σύστημα. Πλέον, μετά και από τις παραπάνω διαδικασίες που περιγράφηκαν υπάρχουν στο σύστημα τα κρυπτογραφημένα υποαρχεία

του αρχικού αρχείου. Επομένως, πρώτα θα πρέπει να πραγματοποιηθεί η αποκρυπτογράφηση των υποαρχείων. Για τον σκοπό αυτό χρησιμοποιήθηκε η συνάρτηση decryptMessage που παρουσιάζεται στο σχήμα 4.5 και η οποία δέχεται ως όρισμα το αναγνωριστικό του enclave το οποίο έχει δημιουργηθεί επιτυχώς, τα κρυπτογραφημένα δεδομένα, το μέγεθος των κρυπτογραφημένων δεδομένων, έναν δείκτη στην δομή στην οποία θα αποθηκευτούν τα αποκρυπτογραφημένα δεδομένα και τέλος το μέγεθος της δομής που θα χρησιμοποιηθεί. Όπως αντίστοιχα συμβαίνει και με την διαδικασία της κρυπτογράφησης έτσι και εδώ μετά από κάθε επιτυχή κλήση της παραπάνω συνάρτησης δημιουργείται ένα νέο αρχείο το οποίο αποθηκεύει τα δεδομένα τα οποία αποκρυπτογραφήθηκαν. Μετά το τέλος της παραπάνω διαδικασίας παράγονται υποαρχεία στα οποία πρέπει να γίνει συρραφή ώστε να προκύψει το αρχικό αρχείο. Στην περίπτωση που δεν είχε πραγματοποιηθεί αρχικά διάσπαση του αρχείου το μόνο που απαιτείται είναι έλεγχος για την σωστή κωδικοποίηση του. Δηλαδή θα πρέπει να ελεγχθεί αν χρειάζεται να γίνει αποκωδικοποίηση ώστε να επανέλθει στην αρχική μορφή που παραδόθηκε στο σύστημα.

```
// Decrypt each chunk and save it to the corresponding chunk file
ret = decryptMessage(eid, encrypted_message,
length_of_encrypted_message, decrypted_message,
length_of_decrypted_message);
```

Σχήμα 4.5: Κλήση συνάρτησης C++ για την αποκρυπτογράφηση αρχείου.

4.5.1 Η αποκωδικοποίηση και σύνθεση των αρχείων

Σε περίπτωση που κατά την προηγούμενη διαδικασία έχουν δημιουργηθεί πολλαπλά αρχεία τότε θα πρέπει να γίνει σύνθεση τους ώστε να δημιουργηθεί εκ νέου το αρχικό αρχείο το οποίο είχε παραδοθεί στο σύστημα. Πρώτα όμως πραγματοποιείται έλεγχος για την κωδικοποίηση του αρχείου με παρόμοιο τρόπο που είχε πραγματοποιηθεί μετά την διάσπαση. Αν το αρχείο είχε παραδοθεί σε κωδικοποίηση διαφορετική από base64 τότε θα πραγματοποιηθεί αποκωδικοποίηση και θα επανέλθει στην αρχική κωδικοποίηση που παραδόθηκε.

Μετά από το παραπάνω στάδιο πραγματοποιείται στην συσκευή του χρήστη συρραφή των αρχείων που έχουν προκύψει με αποτέλεσμα να υπάρχει πλέον το αρχικό αρχείο το οποίο παραδόθηκε. Η συρραφή των αρχείων δεν αποτελεί μια διαδικασία με σημαντικό υπολογιστικό κόστος επομένως δεν αποτελεί πρόβλημα να

πραγματοποιηθεί τοπικά στην συσκευή του χρήστη. Συμπερασματικά, έχει περιγραφεί ολόκληρη η διαδικασία κρυπτογράφησης και αποκρυπτογράφησης μαζί με όλες τις υπόλοιπες απαραίτητες διαδικασίες οι οποίες είναι η κωδικοποίηση/αποκωδικοποίηση των αρχείων καθώς και η διάσπαση/σύνθεση τους όταν αυτό είναι απαραίτητο.

4.6 Η ομομορφική κρυπτογράφηση

Ένας ακόμη τρόπος κρυπτογράφησης που εξετάσαμε ήταν αυτός της ομομορφικής κρυπτογράφησης. Σκοπός είναι να παράξουμε ένα κρυπτογραφημένο αρχείο με ένα μυστικό κλειδί το οποίο παράγεται τοπικά. Με το ίδιο κλειδί πραγματοποιείται η αποκρυπτογράφηση ώστε εν τέλει να ανακτηθεί το αρχικό αρχείο. Για την πραγματοποίηση των παραπάνω διαδικασιών χρησιμοποιήθηκε η βιβλιοθήκη ανοιχτού κώδικα HElib[?] η οποία έχει αναπτυχθεί από στα πλαίσια έρευνας της εταιρείας IBM και έχει βελτιωθεί με την συμβολή και άλλων ερευνητικών ομάδων. Η συγκεκριμένη βιβλιοθήκη καθώς και ο κώδικας ο οποίος αναπτύχθηκε για την χρήση της είναι σε γλώσσα C++. Επιπλέον πραγματοποιήθηκε και πάλι έλεγχος και διόρθωση στις περιπτώσεις τις οποίες κάποιο αρχείο δεν είχε την απαραίτητη κωδικοποίηση base64. Όπως αναφέραμε και πριν σκοπός της συγκεκριμένης υλοποίησης είναι η σύγκριση της απόδοσης της με αυτήν που αξιοποιεί το SGX.

4.6.1 Η δημιουργία του μυστικού κλειδιού

Για την δημιουργία του μυστικού κλειδιού χρησιμοποιήσαμε η βιβλιοθήκη cryptography [?] της Python. Σκοπός ήταν η παραγωγή ενός τυχαίου ασύμμετρου κλειδιού μεγέθους 128 bit διαφορετικό για κάθε αρχείο που κρυπογραφείται. Το κλειδί μπορεί να έχει μεταβλητό μέγεθος. Μετά την δημιουργία του κλειδιού μπορεί να πραγματοποιηθεί η διαδικασία της ομομορφικής κρυπτογράφησης.

4.6.2 Κρυπτογράφηση με την βιβλιοθήκη HElib

Πριν πραγματοποιηθεί η κρυπτογράφηση ενός αρχείου είναι απαραίτητο να γίνουν ορισμένες αρχικοποιήσεις. Πιο συγκεκριμένα, οι αρχικοποιήσεις αυτές αφορούν δομές οι οποίες είναι απαραίτητες για την υλοποίηση του σχήματος BGV (Brakerski-Gentry-Vaikuntanathan) που χρησιμοποιεί η βιβλιοθήκη. Στην συνέχεια αφού γίνει

η ανάγνωση των δεδομένων του αρχείου καθώς και του μυστικού κλειδιού που έχει παραχθεί πραγματοποιείται ομομορφική κρυπτογράφηση. Ακολούθως μπορεί να πραγματοποιηθεί μια πράξη μεταξύ δεδομένων που έχουν κρυπτογραφηθεί με το ίδιο κλειδί. Τα κρυπτογραφημένα δεδομένα αποθηκεύονται σε αρχεία με αντίστοιχο τρόπο που έγινε και με την χρήση Intel SGX.

Η βιβλιοθήκη HElib παρέχει κλήση συνάρτησης με όνομα encrypt η οποία φαίνεται στο σχήμα 4.6 και δέχεται ως όρισμα τα δεδομένα, το κλειδί και τον χώρο στον οποίο θα αποθηκευτούν τα κρυπτογραφημένα δεδομένα. Μετά την κλήση της συνάρτησης τα κρυπτογραφημένα δεδομένα βρίσκονται στην δομή ciphertext.

```
// Encrypt the plaintext by using the public key
encrypted_array.encrypt(ciphertext, public_key, plaintext);
```

Figure 4.6: Κλήση συνάρτησης ομομορφικής κρυπτογράφησης.

Φυσικά η παραπάνω συνάρτηση είναι υψηλού επιπέδου και υλοποιείται από κώδικα χαμηλότερου επιπέδου στον οποίο έγιναν ορισμένες βελτιώσεις με σκοπό την βελτίωση της απόδοσης. Στο σχήμα 4.7 παρουσιάζεται τμήμα κώδικα που υλοποιεί την ομομορφική κρυπτογράφηση.

```
for (size_t i=0; i<ctxt.parts.size(); i++) {  // add noise to all the parts
  ctxt.parts[i] *= r;
  xdouble e_bound;

  if (highNoise && i == 0) {
    // we sample e so that coefficients are uniform over

    ZZ B;
    B = context.productOfPrimes(context.ctxtPrimes);
    B /= (ptxtSpace*8);

    e_bound = e.sampleUniform(B);
  }
  else {
    e_bound = e.sampleGaussianBounded(stdev);
  }

  e *= ptxtSpace;
  e_bound *= ptxtSpace;

  if (i == 1) {
    e_bound *= getSKeyBound(ctxt.parts[i].skHandle.getSecretKeyID());
  }

  ctxt.parts[i] += e;
  ctxt.noiseBound += e_bound;
}
```

Figure 4.7: Τμήμα κώδικα υλοποίησης ομομορφικής κρυπτογράφησης.

4.6.3 Αποκρυπτογράφηση με την βιβλιοθήκη HElib

Όταν ζητηθεί η αποκρυπτογράφηση κάποιου αρχείου τότε πραγματοποιείται η αντίστροφη διαδικασία. Θα πρέπει λοιπόν να γίνει αποκρυπτογράφηση των δεδομένων, τα οποία είχα κρυπτογραφηθεί με κλειδί το οποίο είχε παραχθεί από την βιβλιοθήκη, ώστε να να επανέλθουν στην αρχική τους μορφή. Έτσι έχουμε πλέον τα δεδομένα στην μορφή που βρισκόντουσαν πριν εφαρμοστεί η ομομορφική κρυπτογράφηση.

Η βιβλιοθήκη HElib παρέχει κλήση συνάρτησης με όνομα decrypt η οποία δέχεται ως όρισμα τα κρυπτογραφημένα δεδομένα, το κλειδί και τον χώρο στον οποίο θα αποθηκευτούν τα αποκρυπτογραφημένα δεδομένα. Μετά την κλήση της συνάρτησης τα αποκρυπτογραφημένα δεδομένα βρίσκονται στην δομή decrypted_text. Η κλήση αυτής της συνάρτησης παρουσιάζεται στο σχήμα 4.8.

```
// Decrypt the ciphertext
encrypted_array.decrypt(ciphertext, secret_key, decrypted_text);
```

Σχήμα 4.8: Κλήση συνάρτησης C++ για την αποκρυπτογράφηση αρχείου.

4.7 Συμμετρική κρυπτογράφηση με αξιοποίηση του υλικού

Εκτός από τις δύο παραπάνω υλοποιήσεις κρίθηκε χρήσιμο να χρησιμοποιηθεί και μια μέθοδος κρυπτογράφησης η οποία αποτελεί την πιο αποδοτική μορφή που αξιοποιεί και το υπάρχον υλικό. Πιο συγκεκριμένα χρησιμοποιήθηκε η βιβλιοθήκη OpenSSL η οποία θεωρείται ότι παρέχει την καλύτερη υλοποίηση του αλγορίθμου AES η οποία μάλιστα εκμεταλλεύεται και το υλικό προκειμένου να επιταχύνει την διαδικασία και να βελτιωθεί αισθητά η απόδοση.

Η συγκεκριμένη βιβλιοθήκη παρέχει μια υλοποίηση του γνωστού αλγορίθμου συμμετρικής κρυπτογράφησης AES-GCM (Galois Counter Mode). Όπως προαναφέρθηκε χρησιμοποιήθηκαν κλήσεις συστήματος με σκοπό να επιταχυνθεί η διαδικασία της κρυπτογράφησης και της αποκρυπτογράφηση. Οι κλήσεις αυτές εμπεριέχονται στις εντολές AES-NI (Advanced Encryption Standard-New Instructions). Την υποστήριξη αυτών των κλήσεων δεν μπορεί να την κάνει οποιοσδήποτε επεξεργαστής. Η εταιρεία Intel παρέχει μια λίστα με τους επεξεργαστές οι οποίοι υποστηρίζουν αυτές τις εντολές. Φυσικά, ο επεξεργαστής που χρησιμοποιήθηκε υποστήριζε όλες τις εντολές συστήματος τύπου AES-NI.

Για την κρυπτογράφηση κάθε block δεδομένων χρησιμοποιήθηκε η συνάρτηση EVP_EncryptUpdate(). Όπως θα δειχθεί και στο επόμενο κεφάλαιο της αξιολόγησης η συγκεκριμένη υλοποίηση είναι τάξεις μεγέθους πιο γρήγορη σε σχέση με τις προηγούμενες δύο υλοποιήσεις. Μια ουσιώδης διαφοροποίηση όμως είναι ότι στην συγκεκριμένη περίπτωση η αξιοποίηση του υλικού γίνεται για λόγους απόδοσης και όχι για να υπάρξει μια επιπλέον εγγύηση ασφάλειας όπως στην περίπτωση του Intel SGX.

Όπως αναφέραμε χρησιμοποιείται υλικό για την κρυπτογράφηση επομένως αξιοποιούνται εντολές assembly για την υλοποίηση της κρυπτογράφησης και της αποκρυπτογράφησης δεδομένων. Στο σχήμα 4.9 παρουσιάζουμε κομμάτι κώδικα assembly των εντολών AES-NI (Advanced Encryption Standard-New Instructions).

```
encrypt:
    # Try to read a block of plaint
    # Exit on EOF.
    call read_block
    cmp  $16, %rax
    jl   exit

    # Encrypt the block.
    pxor       %xmm5,  %xmm0
    aesenc     %xmm6,  %xmm0
    aesenc     %xmm7,  %xmm0
    aesenc     %xmm8,  %xmm0
    aesenc     %xmm9,  %xmm0
    aesenc     %xmm10, %xmm0
    aesenc     %xmm11, %xmm0
    aesenc     %xmm12, %xmm0
    aesenc     %xmm13, %xmm0
    aesenc     %xmm14, %xmm0
    aesenclast %xmm15, %xmm0

    # Write it to stdout and loop.
    call write_block
    jmp  encrypt
```

Figure 4.9: Τμήμα κώδικα assembly που υλοποιεί κρυπτογράφηση δεδομένων.

Chapter 5

Αξιολογηση

5.1 Αξιολόγηση

Εφόσον έχει παρουσιαστεί ο σχεδιασμός και η υλοποίηση κάθε τρόπου κρυπτογράφησης μπορεί να γίνει η αξιολόγηση των υλοποιήσεων. Θα γίνει αναφορά τόσο σε ποιοτικά όσο και σε ποσοτικά χαρακτηριστικά. Πρώτα θα γίνει η ποιοτική αξιολόγηση.

5.2 Ποιοτική αξιολόγηση

Στόχος της ποιοτικής αξιολόγησης είναι να εξεταστεί αν επιτεύχθηκαν οι αρχικοί στόχοι που είχαν τεθεί. Ο βασικότερος στόχος ήταν να μπορεί το σύστημα να υποδεχθεί ένα αρχείο και στην συνέχεια να το κρυπτογραφήσει με σκοπό να εκμεταλλευθεί τις εγγυήσεις ασφάλειας που μπορεί να προσφέρει το υπάρχον υλικό και πιο συγκεκριμένα η τεχνολογία Intel SGX. Επιπροσθέτως, ένας ακόμη βασικός στόχος της παρούσας εργασίας ήταν να γίνει κρυπτογράφηση των δεδομένων με χρήση λογισμικού ομομορφικής κρυπτογράφησης η οποία αποτελεί έναν ανερχόμενο και

πολλά υποσχόμενο τρόπο προστασίας και ταυτόχρονης επεξεργασίας δεδομένων. Τέλος, στόχος ήταν να υλοποιηθεί και μια μέθοδος συμμετρικής κρυπτογράφησης, και πιο συγκεκριμένα να υλοποιηθεί ο αλγόριθμος AES με υποβοήθηση του υπάρχοντος υλικού για λόγους βέλτιστης απόδοσης. Σε ένα ολοκληρωμένο σύστημα νέφους θα μπορούσε να υιοθετηθεί η λύση που προτείνουμε στην σχεδίαση η οποία προβλέπει επικοινωνία του χρήστη με το SGX με χρήση του πρωτοκόλλου TLS. Φυσικά, και άλλες λύσεις μπορούν να προταθούν. Συμπερασματικά, έχουν υλοποιηθεί επιτυχώς όλοι οι παραπάνω τρόποι κρυπτογράφησης που περιγράφηκαν και σε επόμενη υποενότητα θα παρουσιαστούν τα πειραματικά αποτελέσματα. Πρώτα ας δούμε και ας αξιολογήσουμε ξεχωριστά τις δύο βασικές οντότητες του συστήματος.

5.2.1 Ο χρήστης

Ο χρήστης είναι η οντότητα που ενδιαφέρεται για την ασφάλεια των αρχείων του. Όπως έχει αναφερθεί για να γίνει η κρυπτογράφηση απαιτείται τα δεδομένα να έχουν κωδικοποίηση τύπου base64. Υπήρξε πρόβλεψη ακόμη και αν δεν είχαν τα δεδομένα την σωστή κωδικοποίηση καθώς τότε το σύστημα αναλάμβανε μόνο του να τα φέρει στην σωστή κωδικοποίηση.

5.2.2 Το σύστημα

Το σύστημα είναι αυτό το οποίο λαμβάνει τα αρχεία. Είναι υπεύθυνο για την κρυπτογράφηση και συνολικά για την διαχείριση τους. Παρέχει δυνατότητα για κρυπτογράφηση με χρήση Intel SGX, ομομορφικής μεθόδου ή συμμετρική κρυπτογράφηση με εκμετάλλευση υλικού. Συμπερασματικά, παρέχει εγγυήσεις ασφάλειας για τα δεδομένα οι οποίες απορρέουν είτε από την χρήση εξειδικευμένου υλικού είτε από την χρήση μεθόδων λογισμικού που έχουν περιγραφεί.

5.2.3 Συμπεράσματα ποιοτικής αξιολόγησης

Αρχικά, επιτεύχθηκαν οι βασικοί στόχοι της σχεδίασης που ήταν η επιτυχής αλλά και αποδοτική υλοποίηση των κρυπτογραφικών μεθόδων που έχουν περιγραφεί. Επιπλέον, αντιμετωπίστηκε με επιτυχία το ζήτημα της ορθής κωδικοποίησης το οποίο προέκυψε στην διαδικασία της υλοποίησης. Τέτοια ζητήματα είναι λογικό να προκύπτουν χωρίς μάλιστα να έχουν προβλεφθεί στην φάση της σχεδίασης γεγονός

που δείχνει και τις ιδιαιτερότητες της δημιουργίας ενός συστήματος ασφάλειας.

5.3 Ποσοτική αξιολόγηση

Σε αυτήν την ενότητα θα παρουσιαστούν τα πειραματικά αποτελέσματα που αφορούν την απόδοση των τριών υλοποιήσεων που πραγματοποιήθηκαν. Πριν παρουσιαστούν τα αποτελέσματα θα πρέπει να αναφερθούν τα τεχνικά χαρακτηριστικά του συστήματος στο οποίο πραγματοποιήθηκαν όλα τα πειράματα.

5.3.1 Τεχνικά χαρακτηριστικά του συστήματος

Παρακάτω παρουσιάζονται τα τεχνικά χαρακτηριστικά του συστήματος:

- Λειτουργικό Σύστημα: Debian GNU/Linux 10 (buster)
- Επεξεργαστής (CPU): Intel(R) Core(TM) i7-8700K CPU @ 3.70GHz
- Μνήμη τυχαίας προσπέλασης (RAM): 16GiB/64 bits/DDR4 Synchronous 2666 MHz
- Αποθηκευτικός χώρος: ATA Disk 1TB

5.3.2 Καταγραφή του χρόνου κρυπτογράφησης για διαφορετικά μεγέθη αρχείων

Το πρώτο πείραμα αφορούσε τον χρόνο που απαιτείται για να κρυπτογραφηθούν αρχεία διαφορετικών μεγεθών και τα αποτελέσματα παρουσιάζονται στο σχήμα 5.1. Τα πειράματα αφορούν τις τρεις διαφορετικές υλοποιήσεις που έγιναν. Η πρώτη είναι με χρήση Intel SGX, η δεύτερη με ομομορφική κρυπτογράφηση και η τρίτη αφορά κρυπτογράφηση με χρήση αλγορίθμου AES με εκμετάλλευση του υπάρχοντος υλικού. Εξετάστηκαν τα μεγέθη 512KB, 1MB, 64MB, 128MB, 512MB και 1GB . Και στις τρεις υλοποιήσεις το μέγεθος του συμμετρικού κλειδιού ήταν 128bit. Για κάθε μέγεθος και για κάθε υλοποίηση έγιναν 10 διαφορετικές επαναλήψεις. Πιο συγκεκριμένα, χρονομετρήθηκε σε κλίμακα δευτερολέπτων ο συνολικός χρόνος που απαιτείται για την κρυπτογράφηση κάθε αρχείου. Παρουσιάζονται στο επόμενο γράφημα οι μέσοι όροι για τους χρόνους κρυπτογράφησης για κάθε υλοποίηση και

μέγεθος αρχείου. Η κλίμακα του χρόνου είναι λογαριθμική καθώς παρουσιάζεται μεγάλη απόκλιση μεταξύ των χρόνων που απαιτούνται.

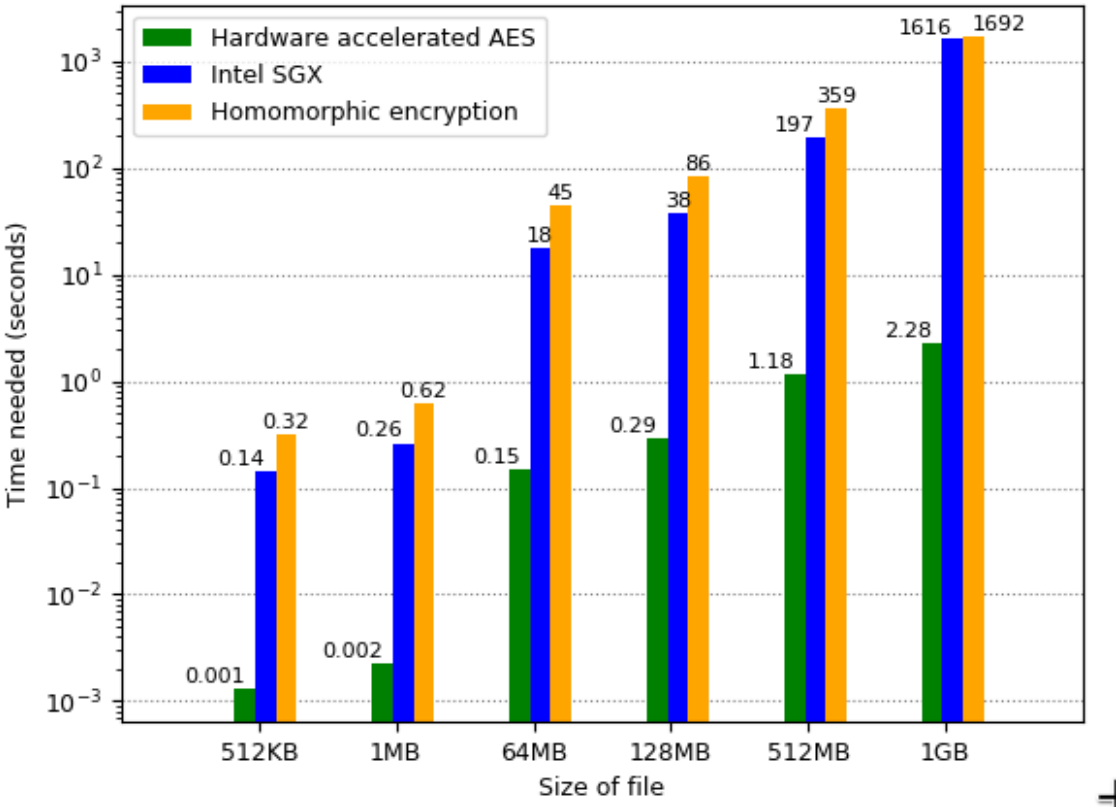

Figure 5.1: Μέσοι χρόνοι κρυπτογράφησης.

Όπως παρατηρούμε η υλοποίηση του αλγορίθμου συμμετρικής κρυπτογράφησης AES είναι τάξεις μεγέθους αποδοτικότερος από τις δύο άλλες υλοποιήσεις. Αυτό είναι ένα αναμενόμενο αποτέλεσμα καθώς η συγκεκριμένη υλοποίηση θεωρείται ως η πιο αποδοτική από άποψη ταχύτητας χωρίς ωστόσο να υπονομεύει την ασφάλεια των δεδομένων.

Συγκρίνοντας τώρα την υλοποίηση της κρυπτογράφησης με χρήση Intel SGX με την υλοποίηση ομομορφικής κρυπτογράφησης παρατηρούμε ότι το SGX είναι αποδοτικότερο. Αυτό είναι ένα αναμενόμενο αποτέλεσμα αν αναλογιστεί κανείς την μαθηματική πολυπλοκότητα που έχουν οι μαθηματικές πράξεις της ομομορφικής κρυπτογράφησης. Η πολυπλοκότητα προκύπτει από τον συνδυασμό του κρυπτογραφημένου κειμένου με το κρυπτογραφημένο συμμετρικό κλειδί. Παρατηρούμε επίσης πως υπάρχει αξιοπρόσεκτη μείωση της απόδοσης της υλοποίησης με Intel SGX για μεγέθη μεγαλύτερα από 128MB. Το τελευταίο συμπέρασμα θα γίνει ευκο-

λότερα κατανοητό από το επόμενο σχήμα το οποίο παρουσιάζει την ρυθμαπόδοση των υλοποιήσεων.

5.3.3 Καταγραφή της ρυθμαπόδοσης της κρυπτογράφησης κάθε υλοποίησης για διαφορετικά μεγέθη αρχείων

Ένας ακόμη τρόπος να εξετάσουμε την απόδοση των τριών υλοποιήσεων είναι να τις συγκρίνουμε ως προς την ρυθμαπόδοση που επιτυγχάνουν για διαφορετικά μεγέθη αρχείων τα οποία κρυπτογραφούν. Ως ρυθμαπόδοση ορίζεται τα Megabyte δεδομένων ανά δευτερόλεπτο τα οποία κρυπτογραφούνται σε κάθε περίπτωση από τις τρεις υλοποιήσεις. Η κλίμακα και σε αυτήν την περίπτωση είναι λογαριθμική καθώς η διαφορά της υλοποίησης του αλγορίθμου AES σε σχέση με τις άλλες υλοποιήσεις παρουσιάζει μεγάλη απόκλιση. Ακολουθεί λοιπόν το σχήμα 5.2 που παρουσιάζει την ρυθμαπόδοση των υλοποιήσεων.

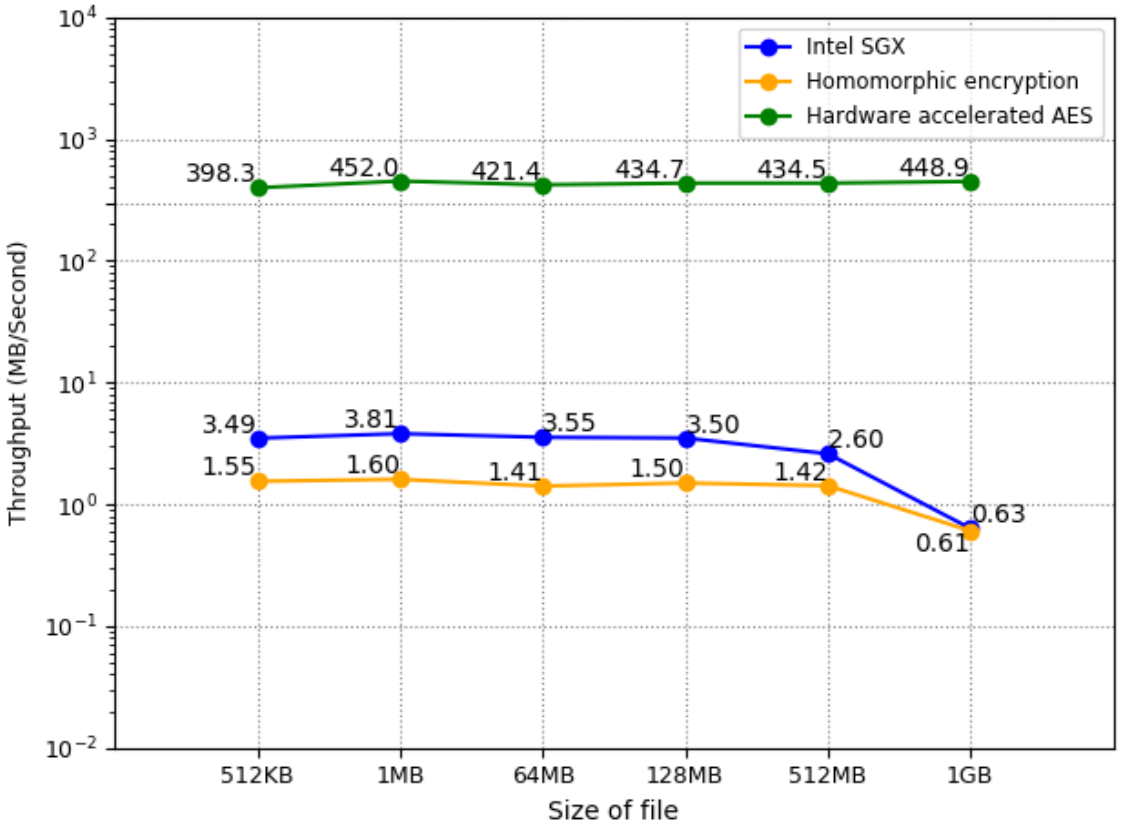

Figure 5.2: Ρυθμαπόδοση κρυπτογράφησης.

Αρχικά, παρατηρούμε την μεγάλη απόκλιση που έχει η ρυθμαπόδοση του αλγορίθμου AES με εκμετάλλευση του υλικού σε σχέση με τις άλλες δύο υλοποιήσεις. Αξιοσημείωτο είναι επίσης ότι η ρυθμαπόδοση με χρήση Intel SGX είναι σταθερά καλύτερη από εκείνη της ομομορφικής κρυπτογράφησης. Ωστόσο, υπάρχει και για τις δύο περιπτώσεις μια μεγάλη πτώση όταν το μέγεθος του αρχείου υπερβαίνει τα 512MB. Οι λόγοι που υπάρχει αυτή η πτώση είναι το μέγεθος της μνήμης του Intel SGX στην μία περίπτωση και η μαθηματική πολυπλοκότητα των απαιτούμενων πράξεων στην περίπτωση της ομομορφικής κρυπτογράφησης.

5.3.4 Η ρυθμαπόδοση της κρυπτογράφησης με περιθώριο λάθους

Ενδιαφέρον είναι να παρουσιάσουμε την ρυθμαπόδοση των υλοποιήσεων λαμβάνοντας υπόψιν ένα διάστημα εμπιστοσύνης (confidence Interval 95%). Στο ραβδόγραμμα και παρουσιάζεται στο σχήμα 5.3 που ακολουθεί παρουσιάζεται η ρυθμαπόδοση των τριών υλοποιήσεων συμπεριλαμβανομένου και του ανάλογου περιθωρίου σφάλματος που προκύπτει από το διάστημα εμπιστοσύνης.

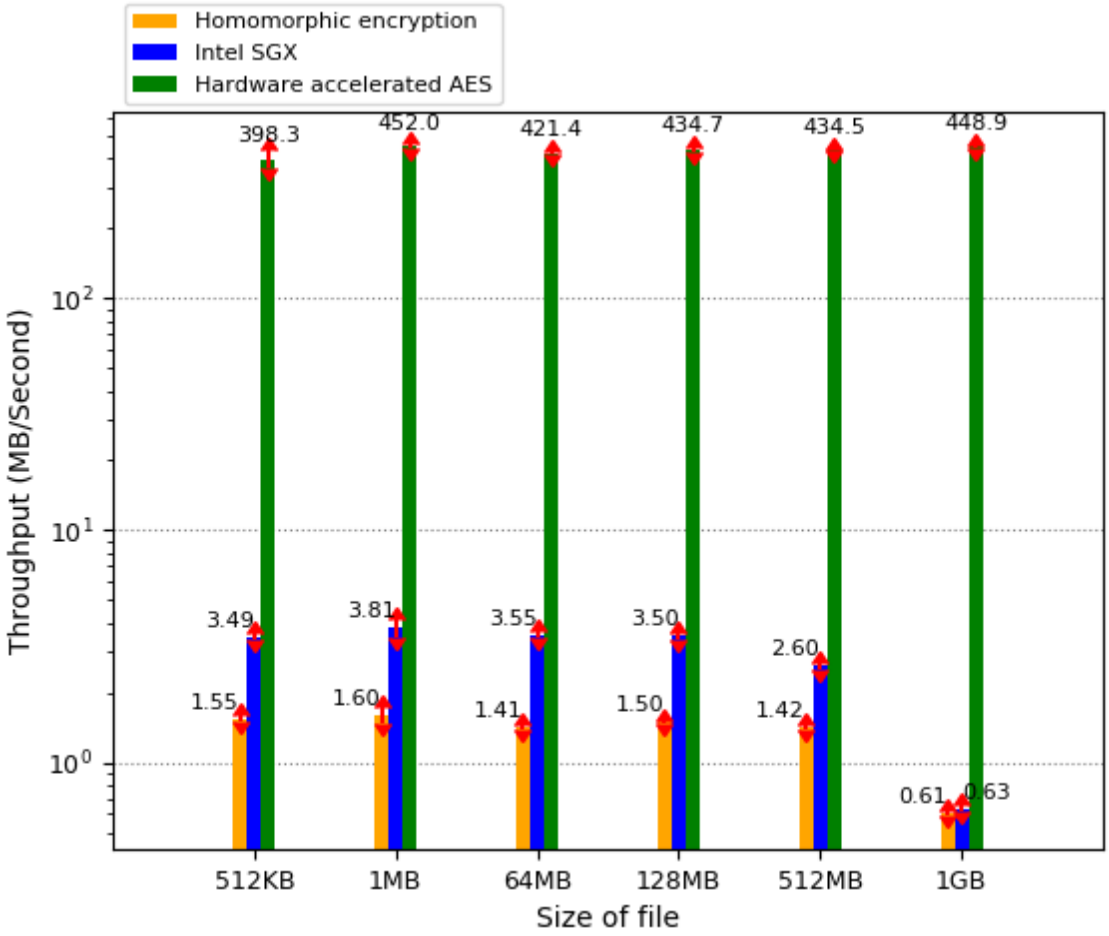

Figure 5.3: Ραβδόγραμμα κρυπτογράφησης με περιθώριο σφάλματος.

Όπως παρατηρούμε δεν υπάρχει μεγάλο περιθώριο σφάλματος γεγονός που οφείλεται στην σχετικά σταθερή απόδοση που είχαν τα πειράματα. Αυτό είναι ένα θετικό στοιχείο και για τις τρεις υλοποιήσεις καθώς εκτός από την καλή απόδοση απαιτείται και σταθερότητα.

5.3.5 Ανάλυση ευαισθησίας (Sensitivity analysis)

Σε αυτήν την υποενότητα θα δούμε την ευαισθησία που έχουν οι υλοποιήσεις της κρυπτογράφησης συμμετρικού κλειδιού AES με εκμετάλλευση υλικού και της ομομορφικής κρυπτογράφησης ως προς το μέγεθος του συμμετρικού κλειδιού. Επαναλήφθηκαν τα πειράματα με τον ίδιο αριθμό πειραμάτων και με τα ίδια μεγέθη αρχείων με μόνη διαφορά το μέγεθος του συμμετρικού κλειδιού. Όπως έχουμε αναφέρει στα προηγούμενα πειράματα το μέγεθος του συμμετρικού κλειδιού ήταν 128

bit. Τα πειράματα λοιπόν επαναλήφθηκαν για μέγεθος συμμετρικού κλειδιού 256 bit. Σκοπός είναι να δούμε κατά πόσο μειώνεται η απόδοση των δυο υλοποιήσεων. Δεν ήταν δυνατό να πραγματοποιηθεί το ίδιο πείραμα και για το Intel SGX καθώς το κλειδί σε αυτήν την περίπτωση έχει σταθερό μέγεθος το οποίο δεν μπορεί να μεταβληθεί. Ακολουθεί πρώτα το διάγραμμα που παρουσιάζεται στο σχήμα 5.4 για την υλοποίηση με συμμετρική κρυπτογράφηση AES με εκμετάλλευση υλικού.

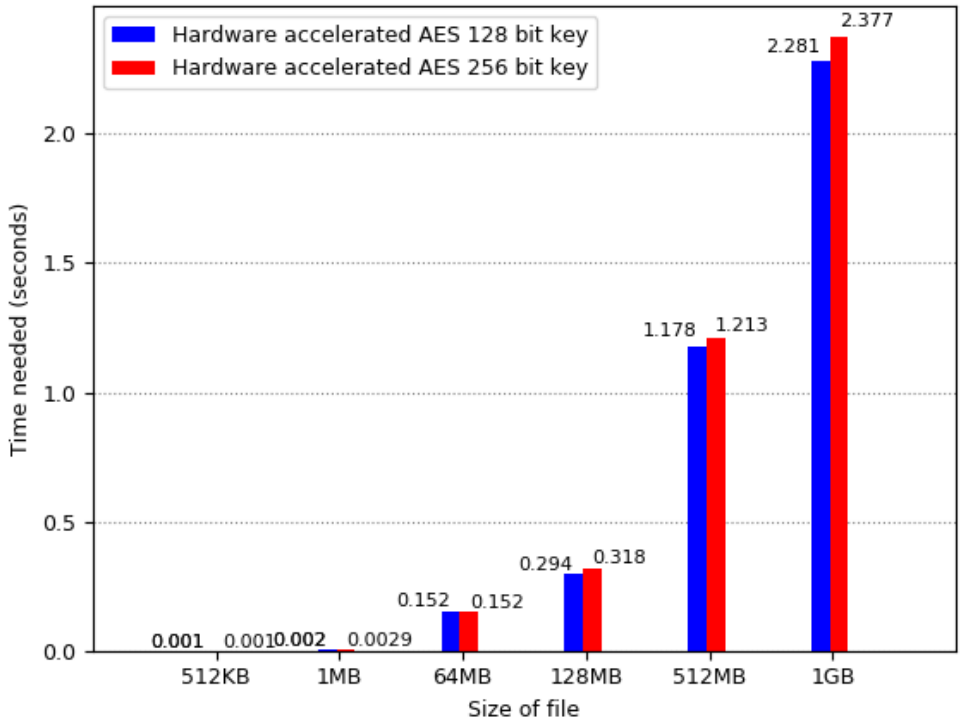

Figure 5.4: Ευαισθησία AES ως προς το μέγεθος του συμμετρικού κλειδιού.

Παρατηρούμε ότι στην συγκεκριμένη υλοποίηση αν και αυξάνεται λίγο ο μέσος χρόνος που χρειάζεται για να κρυπτογραφηθούν τα αρχεία, δεν υπάρχει κάποια αξιοσημείωτη αύξηση ώστε να αλλάζει ουσιαστικά η απόδοση της υλοποίησης. Αυτό είναι ένα θετικό αποτέλεσμα το οποίο αναδεικνύει την σταθερή απόδοση της υλοποίησης. Επομένως η υλοποίηση μπορεί να υποστηρίξει εξίσου αποδοτικά και συμμετρικό κλειδί μεγέθους 256 bit.

Επιπλέον έχει πραγματοποιηθεί αντίστοιχη ανάλυση ευαισθησίας και για την ομομορφική κρυπτογράφηση η οποία διακρίνεται στο σχήμα 5.5.

Όσον αφορά την ομομορφική κρυπτογράφηση παρατηρούμε ότι με την αύξηση του μεγέθους του συμμετρικού κλειδιού παρατηρείται και αύξηση στον μέσο χρόνο

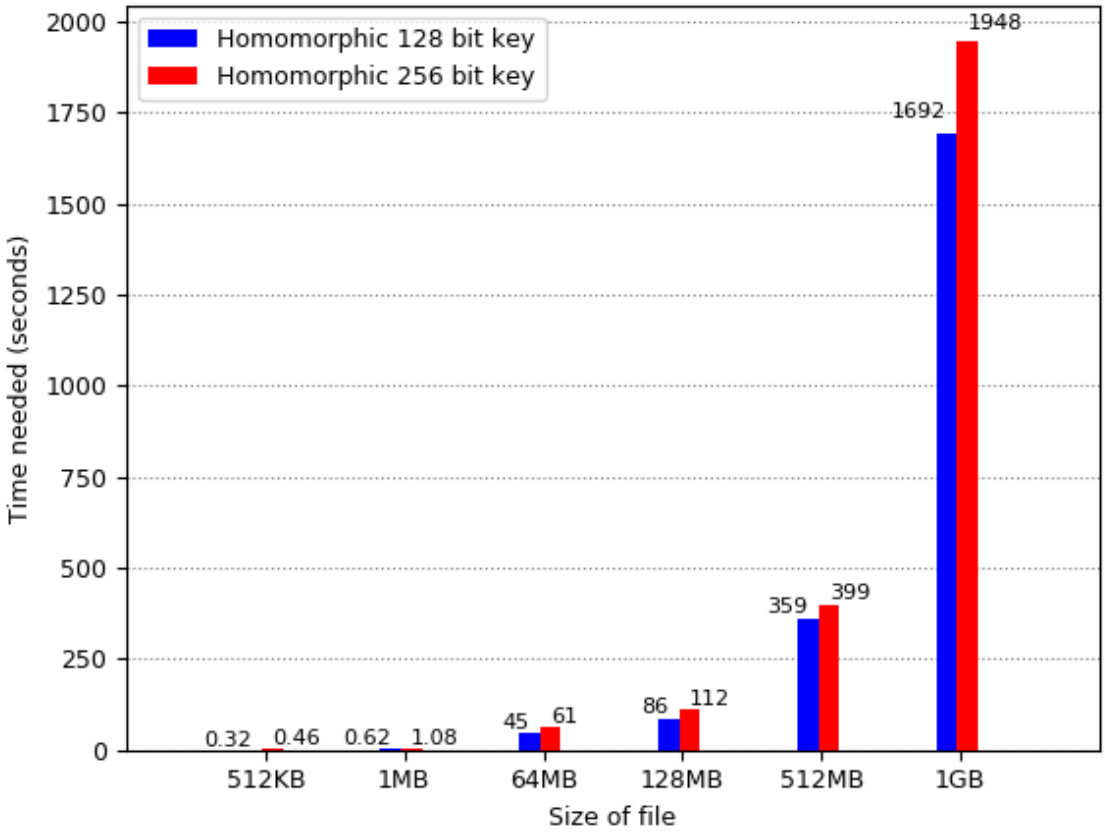

Figure 5.5: Ευαισθησία ομομορφικής κρυπτογράφησης ως προς το μέγεθος του συμμετρικού κλειδιού.

κρυπτογράφησης. Η αύξηση δεν είναι καθοριστική σημασίας αλλά παραμένει αξιοσημείωτη. Συμπερασματικά, η ομομορφική κρυπτογράφηση εμφανίζει ευαισθησία ως προς το μέγεθος του συμμετρικού κλειδιού.

5.3.6 Καταγραφή του χρόνου αποκρυπτογράφησης για διαφορετικά μεγέθη αρχείων

Φυσικά, πέρα από την διαδικασία της κρυπτογράφησης πραγματοποιήθηκε, χρονομετρήθηκε και αξιολογήθηκε η διαδικασία της αποκρυπτογράφησης των τριών υλοποιήσεων. Στο σχήμα 5.6 παρουσιάζονται οι μέσοι χρόνοι αποκρυπτογράφησης όπως αντίστοιχα παρουσιάστηκαν και οι χρόνοι κρυπτογράφησης των δεδομένων.

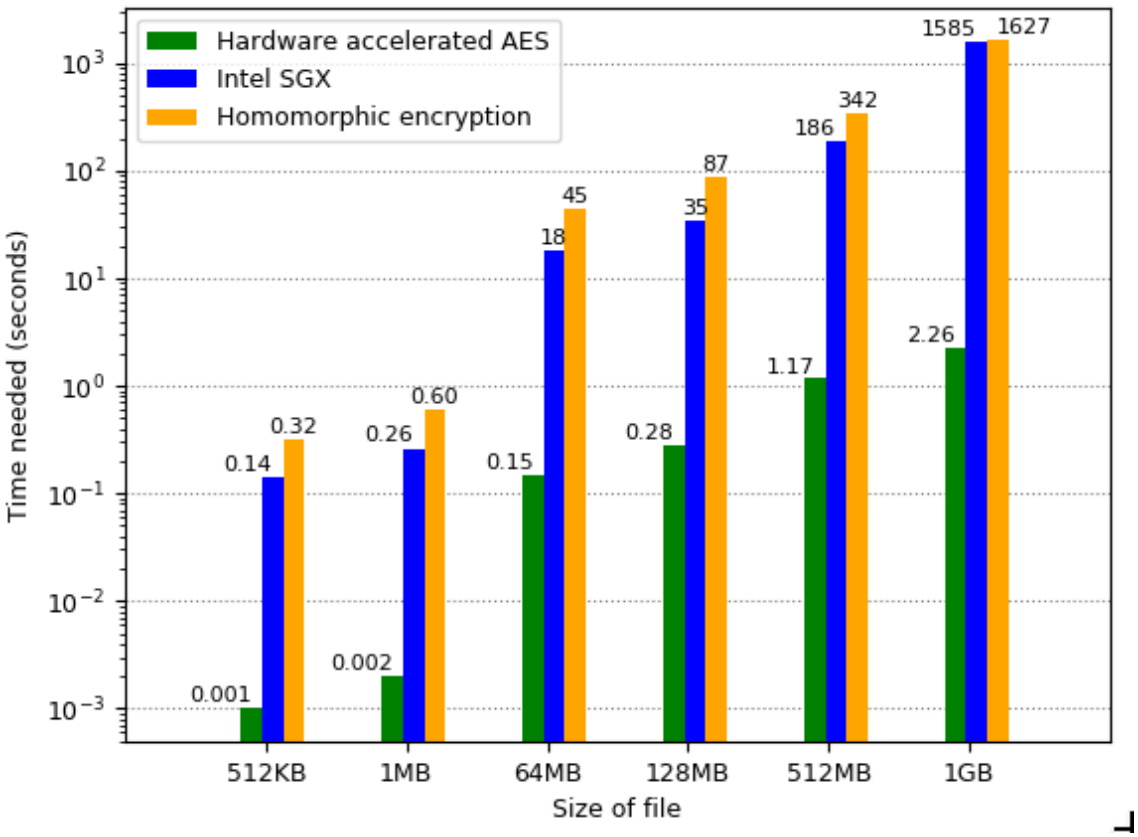

Figure 5.6: Μέσοι χρόνοι αποκρυπτογράφησης.

5.3.7 Καταγραφή της ρυθμαπόδοσης της αποκρυπτογράφησης κάθε υλοποίησης για διαφορετικά μεγέθη αρχείων

Ακόμη ένα σημαντικό γράφημα είναι αυτό της ρυθμαπόδοσης της αποκρυπτογράφησης των δεδομένων για τις τρεις υλοποιήσεις. Παρατηρούμε στο σχήμα 5.7 πως και σε αυτήν την περίπτωση τα αποτελέσματα είναι παρόμοια με την διαδικασία της κρυπτογράφησης δεδομένων.

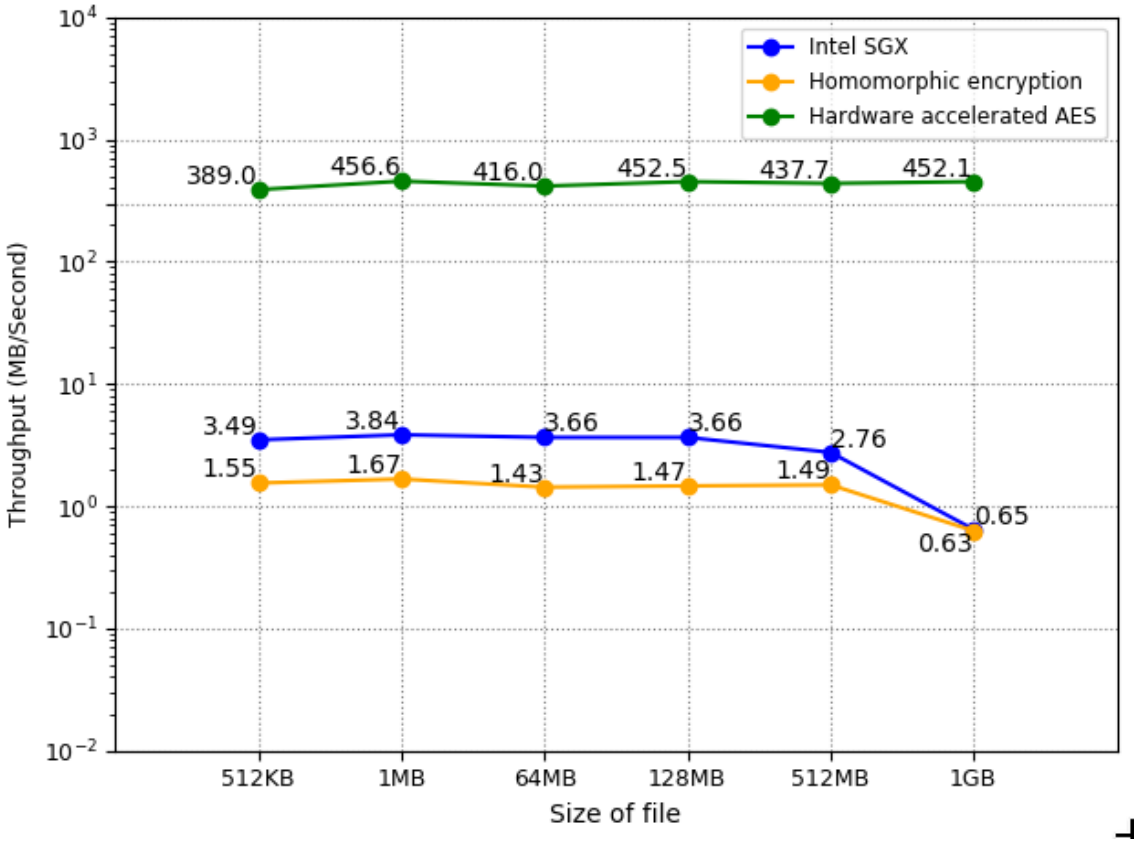

Figure 5.7: Ρυθμαπόδοση αποκρυπτογράφησης.

5.3.8 Η ρυθμαπόδοση της αποκρυπτογράφησης με περιθώριο λάθους

Αντίστοιχα παρουσιάζουμε την ρυθμαπόδοση των υλοποιήσεων για την αποκρυπτογράφηση των δεδομένων λαμβάνοντας υπόψιν ένα διάστημα εμπιστοσύνης (confidence Interval 95%). Στο σχήμα 5.8 που ακολουθεί παρουσιάζεται η ρυθμαπόδοση των τριών υλοποιήσεων συμπεριλαμβανομένου και του ανάλογου περιθωρίου σφάλματος που προκύπτει από το διάστημα εμπιστοσύνης.

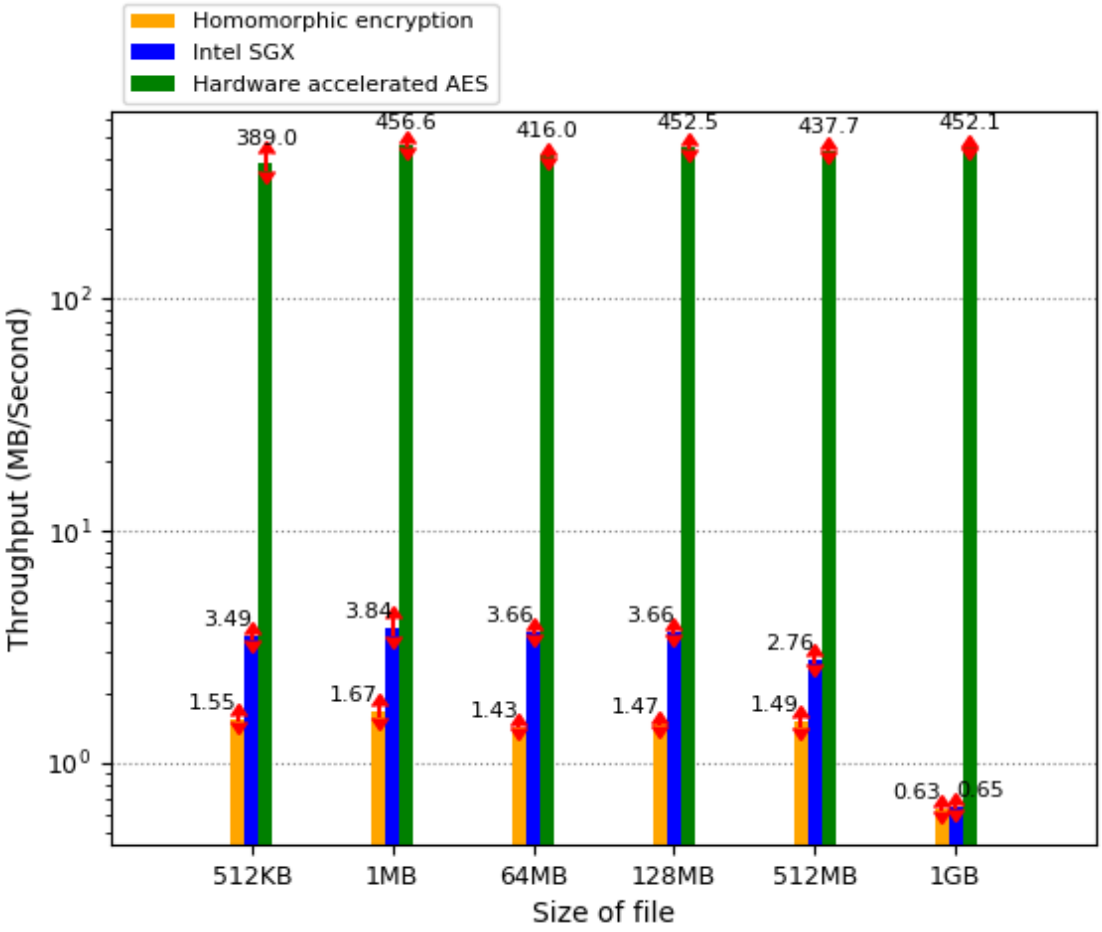

Figure 5.8: Ραβδόγραμμα αποκρυπτογράφησης με περιθώριο σφάλματος.

Chapter 6

Συμπερασματα και μελλοντικη ερευνα

6.1 Συμπεράσματα και μελλοντική έρευνα

Σκοπός της παρούσας μεταπτυχιακής εργασίας ήταν η υλοποίηση και δοκιμή τεχνικών κρυπτογράφησης δεδομένων με χρήση της ανερχόμενης τεχνολογίας Intel SGX. Επίσης, για λόγους σύγκρισης υλοποιήθηκε λύση ομομορφικής κρυπτογράφησης και λύση συμμετρικής κρυπτογράφησης AES με εκμετάλλευση του υπάρχοντος υλικού ώστε να υπάρχει η βέλτιστη δυνατή απόδοση. Επιπροσθέτως, κατά την διάρκεια της υλοποίησης ανιχνεύθηκαν και αντιμετωπίστηκαν ζητήματα που προέκυπταν από την αρχιτεκτονική του υλικού όπως για παράδειγμα το μέγεθος της μνήμης του Intel SGX, είτε ζητήματα όπως η κωδικοποίηση των δεδομένων. Συνεπώς, κατέστη δυνατή η αξιολόγηση και των τριών υλοποιήσεων ως προς την ασφάλεια αλλά και την απόδοση τους. Παρατηρήθηκε, όπως ήταν αναμενόμενο, η απόδοση της υλοποίησης του αλγορίθμου AES με εκμετάλλευση υλικού ήταν τάξης μεγέθους αποδοτικότερη από άποψη χρόνου εκτέλεσης. Σημαντικό γεγονός είναι πως υλοποιήθηκε η κρυπτογράφηση των δεδομένων με χρήση Intel SGX. Με αυτόν τον τρόπο αξιοποιήθηκε μια σχετικά νέα τεχνολογία η οποία βασίζεται αποκλειστικά στο υλικό για να παρέχει εγγυήσεις ασφάλειας. Πιο συγκεκριμένα, ούτε ο διαχειριστής του συστήματος δεν είναι σε θέση να μάθει επακριβώς με ποιόν τρόπο πραγματοποιούνται οι εσωτερικές διεργασίες. Προφανώς δεν είναι και σε θέση να μάθει το συμμετρικό κλειδί,

γεγονός που δίνει μια σημαντική εγγύηση ασφάλειας που δεν είναι δυνατό να δοθεί από λογισμικό.

Το γεγονός ότι παρέχεται εγγύηση ασφάλειας από την σχεδίαση, την αρχιτεκτονική και την λειτουργία του υλικού αποτελεί έναν καινοτόμο και ανερχόμενο τρόπο παροχής εγγυήσεων ασφάλειας. Εξάλλου τα τελευταία χρόνια και άλλες ανταγωνίστριες εταιρείες επενδύουν προς αυτήν την κατεύθυνση η οποία δείχνει να είναι το μέλλον τόσο στην ασφάλεια όσο και στην ιδιωτικότητα των δεδομένων. Αν και υπάρχουν αυτήν την στιγμή περιορισμοί, οι οποίοι και αναφέρθηκαν στην παρούσα εργασία αναμένεται μέσα στα επόμενα χρόνια και στις νέες γενιές επεξεργαστών και γενικότερα νέων προιόντων αυτοί οι περιορισμοί να ξεπεραστούν. Μάλιστα σκοπός είναι να βελτιωθεί η απόδοση τέτοιου είδους τεχνολογιών αλλά και να παρέχονται νέες δυνατότητες για να ικανοποιήσουν νέες ανάγκες που συνεχώς προκύπτουν σε έναν χώρο της σημασίας και της δυναμικής όπως αυτός την ασφάλειας. Άλλωστε σε μια εποχή που η κατάσταση στον τομέα της ασφάλειας συστημάτων, εφαρμογών και της ιδιωτικότητας των δεδομένων είναι τόσο ευμετάβλητη απαιτείται συνεχής αναθεώρηση των υπαρχόντων συστημάτων ασφαλείας και φυσικά η δημιουργία νέων.

Στο μέλλον, με βάση την υπάρχουσα μεταπτυχιακή εργασία θα μπορούσε να γίνει η σχεδίαση ενός ολοκληρωμένου συστήματος, με την κατάλληλη διεπαφή, η οποία θα παρέχει στους χρήστες έναν τρόπο να κρυπτογραφούν τα δεδομένα τους με κάποιο δημόσιο κλειδί το οποίο προκύπτει από το Intel SGX. Επίσης θα μπορούσε να σχεδιαστεί σύστημα για την ασφαλή διαμοίραση δεδομένων σε πολλαπλούς χρήστες η οποία να βασίζεται στην ίδια τεχνολογία. Στην περίπτωση αυτή θα ήταν απαραίτητη η επανακρυπτογράφηση των αρχείων, έννοια η οποία έχει αναφερθεί και προηγουμένως. Πιο συγκεκριμένα, το σύστημα θα μπορούσε να δέχεται ως είσοδο ένα κρυπτογραφημένο αρχείο. Στην συνέχεια ο πάροχος θα αναλαμβάνει να κάνει την επανακρυπτογράφηση πριν στείλει το κρυπτογραφημένο αρχείο σε έναν τελικό χρήστη. Επομένως, για την αλλαγή του κλειδιού κρυπτογράφησης το αρχείο δίνεται ως είσοδος στο SGX μαζί με το καινούριο και το παλιό κλειδί κρυπτογραφημένα. Στο εσωτερικό του SGX τα κλειδιά αποκρυπτογραφούνται και στην συνέχεια χρησιμοποιείται το παλιό κλειδί για να αποκρυπτογραφήσει το κείμενο και το καινούριο κλειδί για να το κρυπτογραφήσει ξανά. Μετά, πάντα εσωτερικά του SGX κρυπτογραφείται το καινούριο κλειδί με το δημόσιο κλειδί του παραλήπτη. Μια ακόμη παραλλαγή που θα μπορούσε να πραγματοποιηθεί αφορά την αποθήκευση

των κρυπτογραφημένων υποαρχείων που προκύπτουν από την διαδικασία κρυπτογράφησης με Intel SGX. Τα υποαρχεία θα μπορούσαν να αποστέλλονται πίσω στον χρήστη μετά την κρυπτογράφηση τους και στην συνέχεια εκείνος να τα επαναπροωθεί στην υπηρεσία όταν επιθυμεί να αποκρυπτογραφηθούν. Συμπερασματικά, οι διαδικασίες που σχεδιάστηκαν και υλοποιήθηκαν μπορούν να αποτελέσουν την βάση για τον σχεδίασμό μιας υπηρεσίας νέφους (Cloud Service) η οποία χρησιμοποιεί υλικό για την παροχή εγγυήσεων ασφάλειας.

Bibliography

[1] Nicole Martin, "How Much Data Is Collected Every Minute Of The Day," http://precog.iiitd.edu.in/people/anupama, 2019.

[2] Centers for Disease Control and Prevention, "Reducing Stigma," https://www.cdc.gov/coronavirus/2019-ncov/daily-life-coping/reducing-stigma.html, 2020.

[3] "Intel Software Guard Extensions ," https://software.intel.com/content/www/us/en/develop/topics/software-guard-extensions.html.

[4] "Amazon simple storage service S3 ," https://aws.amazon.com/s3.

[5] "Dropbox ," https://www.dropbox.com.

[6] "Google cloud storage ," https://cloud.google.com/storage.

[7] Diffie, W. and Hellman, M., "New Directions in Cryptography," *IEEE Trans. Inf. Theor.*, vol. 22, no. 6, 2006. [Online]. Available: https://doi.org/10.1109/TIT.1976.1055638

[8] Sahai, Amit and Waters, Brent, "Fuzzy Identity-Based Encryption," in *Advances in Cryptology – EUROCRYPT 2005*, Cramer, Ronald, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005.

[9] Waters, Brent, "Ciphertext-Policy Attribute-Based Encryption: An Expressive, Efficient, and Provably Secure Realization," in *Public Key Cryptography – PKC 2011*, Catalano, Dario and Fazio, Nelly and Gennaro, Rosario and Nicolosi, Antonio, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 53–70.

[10] Craig Gentry, "A fully homomorphic encryption scheme," Ph.D. dissertation, Stanford University, 2009.

[11] Dowlin, Nathan and Gilad-Bachrach, Ran and Laine, Kim and Lauter, Kristin and Naehrig, Michael and Wernsing, John, "CryptoNets: Applying Neural Networks to Encrypted Data with High Throughput and Accuracy," Tech. Rep. MSR-TR-2016-3, February 2016. [Online]. Available: https://www.microsoft.com/en-us/research/publication/cryptonets-applying-neural-networks-to-encrypted-data-with-high-throughput-and-accurac

[12] Yi, Xun and Paulet, Russell and Bertino, Elisa, *Homomorphic Encryption and Applications*. Springer Publishing Company, Incorporated, 2014.

[13] Syalim, Amril and Nishide, Takashi and Sakurai, Kouichi, "Realizing Proxy Re-encryption in the Symmetric World," vol. 251, 11 2011.

[14] K. Sakurai and T. Nishide and A. Syalim, "Improved proxy re-encryption scheme for symmetric key cryptography," in *2017 International Workshop on Big Data and Information Security (IWBIS)*, 2017, pp. 105–111.

[15] Marty Puranik, "What Is The Cost Of A Data Breach?" https://www.forbes.com/sites/forbestechcouncil/2019/12/02/what-is-the-cost-of-a-data-breach, 2019.

[16] "Trusted Computing Group (TCG)," https://trustedcomputinggroup.org.

[17] Arthur, Will and Challener, David, *A Practical Guide to TPM 2.0: Using the Trusted Platform Module in the New Age of Security*, 1st ed. USA: Apress, 2015.

[18] Rijmen, Vincent and Oswald, Elisabeth, "Update on SHA-1," in *Topics in Cryptology – CT-RSA 2005*, Menezes, Alfred, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 58–71.

[19] Google, *Google Cloud whitepaper*, 2019.

[20] Amazon, *Amazon Web Services: Overview of Security Processes*, 2020.

[21] "Amazon Virtual Private Cloud (VPC)," https://aws.amazon.com/vpc.

[22] Vahldiek-Oberwagner, Anjo and Elnogomi, Eslam and Mehta, Aastha and Garg, Deepak and Druschel, Peter and Rodrigues, Rodrigo and Gehrke, Johannes and Post, Ansley, "Guardat: Enforcing data policies at the storage layer," 04 2015.

[23] "Arm TrustZone," https://developer.arm.com/ip-products/security-ip/trustzone.

[24] "AMD Secure Encrypted Virtualization (SEV)," https://developer.amd.com/sev.

[25] V. Costan and S. Devadas, "Intel SGX Explained," *IACR Cryptol. ePrint Arch.*, vol. 2016, p. 86, 2016.

[26] "Intel Software Guard Extensions (Intel SGX) SDK," https://github.com/intel/linux-sgx.

[27] "Intel Software Guard Extensions (Intel SGX) Developer Guide," https://software.intel.com/content/www/us/en/develop/documentation/sgx-developer-guide.

[28] "HElib library," https://github.com/homenc/HElib.

[29] "Python cryptography," https://cryptography.io.

Appendix A΄

Παραρτημα 4ου Κεφαλαιου

Στο παρόν παράρτημα θα παρουσιαστούν ορισμένα βήματα που αν και είναι ίσως απλοϊκά από άποψη υλοποίησης ήταν απαραίτητα για την ορθή λειτουργία του συστήματος κρυπτογράφησης δεδομένων με χρήση του Intel SGX.

Α΄.1 Η απαραίτητη προεργασία

Δεν θα ήταν σωστό να ξεκινά η διαδικασία της κρυπτογράφησης χωρίς να πραγματοποιηθεί κάποιος έλεγχος για την μορφή και το μέγεθος του αρχείου. Μάλιστα όπως περιγράφηκε και προηγουμένως υπάρχουν διαφορετικές διαδικασίες σε περίπτωση που το μέγεθος του αρχείου δεν είναι διαχειρίσιμο από το SGX. Επιπλέον, θα πρέπει να γίνει και έλεγχος για την μορφή του αρχείου και πιο συγκεκριμένα για την κωδικοποίηση του. Η διαδικασία, η οποία παρουσιάζεται στο σχήμα 4.10, όπως είναι λογικό ξεκινά την στιγμή που ένα αρχείο εμφανίζεται σε έναν συγκεκριμένο φάκελο του συστήματος ο οποίος έχει επιλεχθεί σε προγενέστερη στιγμή από τον κάτοχο του συστήματος. Υπάρχει λοιπόν ένα κατάλληλο bash script το οποίο αναμένει συνεχώς για την εμφάνιση αρχείων στον συγκεκριμένο φάκελο. Επομένως, το συγκεκριμένο bash script έχει τον ρόλο του συντονιστή των ενεργειών.

Figure Α΄.1: Η λειτουργία του script διαχείρησης

Α΄.2 Ο έλεγχος του μεγέθους ενός αρχείου

Στην συνέχεια θα πρέπει να γίνει έλεγχος για το μέγεθος του αρχείου με σκοπό αν είναι παραπάνω από 128MB να υποστεί διάσπαση πριν γίνει η κρυπτογράφηση. Για την πραγματοποίηση της διάσπασης υλοποιήθηκε ρουτίνα σε γλώσσα προγραμματισμού Python. Η συγκεκριμένη ρουτίνα καλείται μετά την εμφάνιση του αρχείου το οποίο θα πρέπει να κρυπτογραφηθεί. Αν το αρχείο έχει μέγεθος μικρότερο από 128MB τότε δεν απαιτείται κάποια διάσπαση καθώς όπως θα δειχθεί και πειραματικά το SGX είναι σε θέση να το διαχειριστεί αποδοτικά.

Αν το μέγεθος είναι μεγαλύτερο τότε το αρχείο δεν είναι διαχειρίσιμο και θα πρέπει να διασπαστεί σε τμήματα των 128MB. Στο σχήμα 4.11 που ακολουθεί φαίνεται το τμήμα του κώδικα το οποίο πραγματοποιεί την διάσπαση ενός αρχείου σε υποαρχεία.

```
while True:
    # Read 128MB chunks of the file
    chunk = file_to_split.read(CHUNK_SIZE)
    if not chunk: break
```

Figure Α΄.2: Κλήση συνάρτησης Python για την διάσπαση ενός αρχείου.

Appendix B′

Παραρτημα 5ου Κεφαλαιου

Για την ποιοτική και την ποσοτική αξιολόγηση που παρουσιάστηκαν απαιτήθηκε να γίνει μια σειρά πειραμάτων. Για την ορθή και ακριβή πραγματοποίηση των πειραμάτων αναπτύχθηκε αντίστοιχος κώδικας για κάθε μια από τις τρεις ξεχωριστές υλοποιήσεις που εξετάστηκαν. Πιο συγκεκριμένα σκοπός ήταν η ακριβής χρονομέτρηση διαδικασιών.

Β′.1 Η χρονομέτρηση στις υλοποιήσεις Intel SGX και ομομορφικής κρυπτογράφησης

Για τις χρονομετρήσεις που απαιτήθηκαν για την κρυπτογράφηση με χρήση Intel SGX και για την ομομορφική κρυπτογράφηση χρησιμοποιήθηκε η συνάρτηση high_resolution_clock::now( ). Η συνάρτηση αυτή επιστρέφει την τρέχουσα χρονική στιγμή που επιστρέφει το ρολόι του συστήματος με ακρίβεια millisecond. Η διαδικασία έχει ως εξής:

- Κλήση της συνάρτησης ακριβώς στο σημείο που ξεκινά η περιοχή του κώδικα που επιθυμούμε να χρονομετρήσουμε.

- Κλήση της συνάρτησης ακριβώς μετά σημείο που ολοκληρώνεται η περιοχή του κώδικα που επιθυμούμε να χρονομετρήσουμε.
- Σύγκριση των δύο τιμών που έχουν προκύψει ώστε να υπολογιστεί ο χρόνος που χρειάστηκε για την εκτέλεση του κώδικα που μας ενδιαφέρει να χρονομετρήσουμε.

Με την παραπάνω διαδικασία λοιπόν είμαστε σε θέση να γνωρίζουμε με ακρίβεια millisecond τον χρόνο που χρειάστηκε για να εκτελεστεί ο κώδικας που μας ενδιαφέρει.

Β΄.2 Η χρονομέτρηση στην υλοποιήση του AES με εκμετάλλευση υλικού

Η κρυπτογράφηση των δεδομένων με τον αλγόριθμο AES υλοποιήθηκε σε γλώσσα C. Ακολουθήσαμε την ίδια διαδικασία αλλά σε αυτήν την περίπτωση χρησιμοποιήσαμε την συνάρτηση clock( ) η οποία μετρά κύκλους επεξεργαστή. Για να μετατραπεί το αποτέλεσμα σε πραγματικό χρόνο πραγματοποιείται διαίρεση με τον αριθμό των κύκλων επεξεργαστή ανά δευτερόλεπτο (CLOCKS_PER_SEC). Αν και σε διαφορετικά συστήματα μπορούν να πραγματοποιηθούν κάποιες αποκλίσεις η ακρίβεια αυτής της μεθόδου βρίσκεται ανάμεσα στο εκατοστό με εκατομμυριοστό του δευτερολέπτου.

Β΄.2.1 Εξαγωγή συμπερασμάτων από τις χρονομετρήσεις

Φυσικά, η ακριβής χρονομέτρηση των διαδικασιών δεν μπορεί να μεταφραστεί σε χρήσιμα συμπεράσματα αν δεν υπάρξει η πρέπουσα αξιοποίηση τους. Για τον λόγο αυτό αναπτύχθηκε κώδικας σε γλώσσα Python, που παρουσιάζεται στο σχήμα 5.9 και ο οποίος αναλαμβάνει να κάνει τους υπολογισμούς που απαιτούνται. Ενδεικτικά αναφέρουμε ορισμένα χρήσιμα στατιστικά που υπολογίσθηκαν:

- Υπολογισμός μέσου όρου.
- Υπολογισμός διαστήματος εμπιστοσύνης (confidence interval).
- Υπολογισμός ρυθμαπόδοσης (throughput).

```
# Calculate the mean of all values
mean = sum_of_experiments/number_of_experiments
print "Mean: ", mean

# Calculate the confidence interval
confidence_interval = st.t.interval(0.95, len(list_of_experiments)-1, loc=np.mean(list_of_experiments), scale=st.sem(list_of_experiments))
print "Confidence interval: ", confidence_interval

# Calculate the throughput
throughput = thr_index/mean
print "Throughput: ", throughput
```

Figure Β΄.1: Κώδικας Python για τον υπολογισμό στατιστικών

ΑΝΑΡΓΥΡΟΣ ΚΑΤΣΟΥΛΙΕΡΗΣ

Ιωάννινα, Ήπειρος
argiris.katsoulieris@gmail.com | 6970049610 | linkedin.com/in/argiris-katsoulieris/

Προγραμματιστής διαδικτυακών εφαρμογών και φοιτητής Μεταπτυχιακού επιπέδου με τίτλο ¨Προηγμένα Υπολογιστικά Συστήματα¨ στο τμήμα Μηχανικών Η/Υ και Πληροφορικής στο Πανεπιστήμιο Ιωαννίνων.

- ★ Εκτενής εμπειρία σε PHP, Python, Ruby-on-Rails, Java, C/C++, HTML5/CSS3, JavaScript, .NET, MongoDB, SQL
- ★ Διάθεση εκμάθησης νέων τεχνολογιών και μεθοδικότητα με έμφαση στην λεπτομέρεια
- ★ Ανάληψη πρωτοβουλιών με σκοπό την άρτια ολοκλήρωση της εκάστοτε εργασίας
- ★ Δυνατότητα τόσο για ανεξάρτητη όσο και για ομαδική εργασία
- ★ Διαρκής εκμάθηση νέων γλωσσών προγραμματισμού και frameworks μέσω διασικτιακών μαθημάτων που παρέχονται από το Udemy και τος Coursera

ΕΡΓΑΣΙΑΚΗ ΕΜΠΕΙΡΙΑ

Προγραμματιστής διαδικτυακών εφαρμογών **ΣΕΠΤΕΜΒΡΙΟΣ 2018–ΙΑΝΟΥΑΡΙΟΣ 2020**

OFIDIA 2 – Τμήμα Μηχανικών Η/Υ και Πληροφορικής, Πολυτεχνική σχολή Πανεπιστημίου Ιωαννίνων

Εργασία ως προγραμματιστής λογισμικού στο έργο "Ofidia 2 - Operational Fire Danger" prevention platform, χρηματοδότηση €1.8M από το ευρωπαικό πρόγραμμα Interreg V-A Greece-Italy 2014-2020.

- Υπεύθυνος σχεδιασμού της λογικής της διαδικτυακής εφαρμογής από την μεριά του διακομιστή καθώς και του σχήματος της βάσης δεδομένων.
- Σχεδιασμός και υλοποίηση λύσεων για την αποθήκευση δεδομένων σε MongoDB και Python
- Σχεδιασμός της αρχιτεκτονικής και του τρόπου καταγραφής των περιστατικών πυρκαγιάς καθώς και σχεδιασμός της ιστοσελίδας καταγραφής
- Βελτιστοποίηση της διεπαφής με σκοπό την ευχρηστία σε κινητές συσκευές
- Ανάπτυξη εργαλείων με σκοπό την αποδοτική και ευχάριστη χρήση όπως διαδραστική σχεδίαση πάνω σε χάρτη

Πρακτική άσκηση **ΙΟΥΛΙΟΣ 2017 – ΑΥΓΟΥΣΤΟΣ 2017**

Athens Voice – Τμήμα διαδικτυακής έκδοσης, Αθήνα

Πρακτική άσκηση κατά τον 4^{ο} χρόνο προπτυχιακών σπουδών

- Δημιουργία νέας ιστοσελίδας για τον ραδιοφωνικό σταθμό της Athens Voice με χρήση Wordpress
- Σχεδιασμός και δημιουργία διαδικτυακών διαφημίσεων στην βασική ιστοσελίδα της εταιρείας

ΕΘΕΛΟΝΤΙΚΗ ΕΡΓΑΣΙΑ

Βοηθός μαθήματος και εργαστηρίου **ΝΟΕΜΒΡΙΟΣ 2017 – ΙΟΥΛΙΟΣ 2020**

Τμήμα Μηχανικών Η/Υ και Πληροφορικής, Πολυτεχνική σχολή Πανεπιστημίου Ιωαννίνων

Διδασκαλία και επίβλεψη φοιτητών σε διάφορα προπτυχιακά μαθήματα (Ασφάλεια υπολογιστών και δικτύων, Λειτουργικά συστήματα, Τεχνολογίες διαδικτύου,)

- Συμμετοχή στον σχεδιασμό του τρόπου διδασκαλίας των μαθημάτων
- Συμμετοχή στον σχεδιασμό των εργαστηριακών ασκήσεων και των διαδικτυακών εργασιών
- Συνεχής παρακολούθηση της προόδου των φοιτητών

ΕΚΠΑΙΔΕΥΣΗ

Μεταπτυχιακό δίπλωμα εξειδίκευσης με θέμα "Προηγμένα Υπολογιστικά Συστήματα'' **ΣΕΠΤΕΜΒΡΙΟΣ 2018 – ΦΕΒΡΟΥΑΡΙΟΣ 2021**

Τμήμα Μηχανικών Η/Υ και Πληροφορικής, Πολυτεχνική σχολή Πανεπιστημίου Ιωαννίνων

Δίπλωμα Ηλεκτρονικού Μηχανικού **ΣΕΠΤΕΜΒΡΙΟΣ 2013 – ΙΟΥΝΙΟΣ 2018**

Τμήμα Μηχανικών Η/Υ και Πληροφορικής, Πολυτεχνική σχολή Πανεπιστημίου Ιωαννίνων