
Πανεπιστημιο Πατρων

Τμήμα Μηχανικών Ηλεκτρονικών Υπολογιστών και Πληροφορικής

---

# Ανάπτυξη Κρυπτογραφικού Πρωτοκόλλου με Χρήση της Κρυπτογράφησης Ελλειπτικών Καμπυλών σε Ασύρματα Δίκτυα Αισθητήρων

---

Διπλωματική Εργασία
του
Απόστολου Πυργελή

*Επιβλέπων Καθηγητής*
Παύλος Σπυράκης

*Συνεπιβλέποντες*
Βασιλική Λιάγκου
Ιωάννης Χατζηγιαννάκης

Πάτρα, Οκτώβριος 2009

# Ευχαριστίες

Για την εκπόνηση της διπλωματικής αυτής εργασίας θα ήθελα να ευχαριστήσω θερμά τον επιβλέποντα καθηγητή κ. Παύλο Σπυράκη καθώς και τους διδάκτορες Βασιλική Λιάγκου και Ιωάννη Χατζηγιαννάκη για την συστηματική υποστήριξη και καθοδήγηση τους.Η συνδρομή τους ήταν καθοριστική σε όλα τα επίπεδα.

Επίσης, επειδή με την εργασία αυτή, ολοκληρώνονται οι σπουδές μου ως προπτυχιακός φοιτητής του Πανεπιστημίου Πατρών θα ήθελα να ευχαριστήσω όλους όσους με βοήθησαν και με στήριξαν τα χρόνια αυτά, ιδιαιτέρως την οικογένεια μου, που με υποστήριξε σε όλες μου τις αποφάσεις με κάθε τρόπο.

# Πρόλογος

Τα ασύρματα δίκτυα αισθητήρων αποτελούν μια νέα κατηγορία δικτύων υπολογιστών. Αποτελούνται από ένα μεγάλο πλήθος υπολογιστικών κόμβων μικροσκοπικού μεγέθους, εφοδιασμένων με πλήθος αισθητήρων και μονάδων ελέγχου. Σκοπός τους είναι η επίτευξη μιας δύσκολης, για τα δεδομένα του κάθε κόμβου, αποστολής μέσω της συνεργασίας μεταξύ όλων των κόμβων του δικτύου. Τα δίκτυα αυτά αντιμετωπίζονται με μεγάλο ενδιαφέρον από την ερευνητική κοινότητα τα τελευταία χρόνια, με αποτέλεσμα να έχουν προταθεί διάφορες πιθανές εφαρμογές τους και αρκετά πρωτόκολλα διάδοσης και επεξεργασίας της πληροφορίας.Παρόλα αυτά, η ασύρματη φύση επικοινωνίας και οι περιορισμένοι πόροι των δικτύων αυτών, τα καθιστά ευάλωτα σε επιθέσεις που στοχεύουν στην διακοπή των λειτουργιών τους.Για το λόγο αυτό, η ασφάλεια στα ασύρματα δίκτυα αισθητήρων είναι αναγκαίο να μελετηθεί περισσότερο, προκειμένου να προστατευθεί η λειτουργία τους.

Σκοπός της παρούσας διπλωματικής εργασίας, είναι η ανάπτυξη ενός πρωτοκόλλου ασφαλούς επικοινωνίας, με χρήση της κρυπτογράφησης ελλειπτικών καμπυλών (ECC: Elliptic Curve Cryptography), σε ασύρματα δίκτυα αισθητήρων.Η κρυπτογράφηση ελλειπτικών καμπυλών, παρέχει διάφορα πλεονεκτήματα που την καθιστούν κατάλληλη για εφαρμογή στις περιορισμένων πόρων συσκευές αισθητήρων.Το πρωτόκολλο αναπτύχθηκε σε πραγματικές συσκευές της σειράς iSense και αξιολογήθηκαν, ως προς το επίπεδο χρονικής καθυστέρησης, οι λειτουργίες δημιουργίας δημοσίου κλειδιού, διανομής κλειδιού και κρυπτογράφησης/αποκρυπτογράφησης δεδομένων με βάση τις ελλειπτικές καμπύλες.

# Περιεχόμενα


**1 Εισαγωγή** **6**

1.1 Ασύρματα Δίκτυα Αισθητήρων . . . . . 6
1.2 Ασφάλεια στα Ασύρματα Δίκτυα Αισθητήρων . . . . . 9
1.3 Αντικείμενο της Διπλωματικής Εργασίας . . . . . 9
1.4 Διάρθρωση της Διπλωματικής Εργασίας . . . . . 10

**2 Σύγχρονη Κρυπτογραφία και Ασφάλεια Συστημάτων** **11**

2.1 Εισαγωγή . . . . . 11
2.2 Ασφάλεια Πληροφορίας και Κρυπτογραφία . . . . . 12
2.3 Τι είναι η Κρυπτογραφία . . . . . 13
2.3.1 Βασική Ορολογία . . . . . 15
2.4 Κρυπτογράφηση Συμμετρικού Κλειδιού . . . . . 19
2.4.1 Block Cipher . . . . . 20
2.4.2 Stream Cipher . . . . . 21
2.5 Κρυπτογράφηση Δημοσίου Κλειδιού . . . . . 21
2.5.1 Η ανάγκη για αυθεντικότητα στα κρυπτοσυστήματα δημοσίου κλειδιού . . . . . 24
2.6 Κρυπτογραφία Συμμετρικού Κλειδιού Εναντίον Κρυπτογραφίας Δημοσίου Κλειδιού . . . . . 25
2.6.1 Πλεονεκτήματα Κρυπτογραφίας Συμμετρικού Κλειδιού . . . . . 26
2.6.2 Μειονεκτήματα Κρυπτογραφίας Συμμετρικού Κλειδιού . . . . . 26
2.6.3 Πλεονεκτήματα Κρυπτογραφίας Δημοσίου Κλειδιού . . . 27
2.6.4 Μειονεκτήματα Κρυπτογραφίας Δημοσίου Κλειδιού . . . 27
2.6.5 Περίληψη της Σύγκρισης . . . . . 28
2.7 Γνωστά Εργαλεία Βασιζόμενα στην Κρυπτογράφηση Δημοσίου Κλειδιού . . . . . 28

2.7.1 RSA Κρυπτογράφηση . . . . . 28
2.7.2 Το σχήμα El Gamal . . . . . 31
2.7.3 Πρωτόκολλο Συμφωνίας Κλειδιών Diffie-Hellman . . . . 33
2.8 Ψηφιακές Υπογραφές . . . . . 36
2.8.1 Ορολογία . . . . . 36
2.8.2 Διαδικασία Υπογραφής . . . . . 37
2.8.3 Διαδικασία Επαλήθευσης . . . . . 37
2.8.4 Απαραίτητες Ιδιότητες των Συναρτήσεων Υπογραφής και Επαλήθευσης . . . . . 38
2.8.5 Ψηφιακές Υπογραφές από Αναστρέψιμη Κρυπτογραφία Δημοσίου Κλειδιού . . . . . 38

**3 Οι Ελλειπτικές Καμπύλες στην Κρυπτογραφία 41**
3.1 Θεωρία Ελλειπτικών Καμπυλών . . . . . 41
3.1.1 Ομάδες Ελλειπτικών Καμπυλών Ορισμένες Πάνω από το Πεδίο των Πραγματικών Αριθμών . . . . . 41
3.1.2 Ομάδες Ελλειπτικών Καμπυλών Ορισμένες Πάνω από τα Πρώτα Πεδία $F_p$ . . . . . 45
3.1.3 Ομάδες Ελλειπτικών Καμπυλών Ορισμένες Πάνω από τα Δυαδικά Πεδία $F_{2^m}$ . . . . . 47
3.2 Οι Ελλειπτικές Καμπύλες στην Σύγχρονη Κρυπτογραφία . . . . 50
3.2.1 Κρυπτογραφία Δημοσίου Κλειδιού και το Πρόβλημα του Διακριτού Λογαρίθμου . . . . . 50
3.2.2 Διάφορα Πρωτόκολλα με Χρήση Ελλειπτικών Καμπυλών 52
3.2.3 Πλεονεκτήματα Ελλειπτικών Καμπυλών . . . . . 56
3.2.4 Βασικές Επιθέσεις σε Κρυπτοσυστήματα Ελλειπτικών Καμπυλών . . . . . 56

**4 Ασφάλεια Σε Ασύρματα Δίκτυα Αισθητήρων 58**
4.1 Εισαγωγή . . . . . 58
4.2 Απειλές και Αρχές Ασφάλειας . . . . . 59
4.2.1 Απειλές . . . . . 59
4.2.2 Αρχές Ασφάλειας . . . . . 61
4.3 Επιθέσεις-Αντίμετρα . . . . . 62
4.3.1 Επιθέσεις στο Φυσικό Επίπεδο . . . . . 62
4.3.2 Επιθέσεις στο Επίπεδο Ζεύξης των Δεδομένων . . . . . 63
4.3.3 Επιθέσεις στα Επίπεδα Δικτύου και Μεταφοράς . . . . . 64
4.4 Κατηγοριοποίηση των Πρωτοκόλλων Ασφαλείας . . . . . 67

**5 iSense: Μια Πλατφόρμα Υλικού και Λογισμικού για Ασύρματα Δίκτυα Αισθητήρων 71**
5.1 Εισαγωγή . . . 71
5.2 Επισκόπηση του Υλικού . . . 71
5.3 Επισκόπηση του Λογισμικού . . . 75
5.4 Πλεονεκτήματα της Πλατφόρμας iSense . . . 78

**6 Ανάπτυξη Πρωτοκόλλου Διαχείρισης Δημοσίου Κλειδιού με Ελλειπτικές Καμπύλες στην Πλατφόρμα iSense 80**
6.1 Εισαγωγή . . . 80
6.2 Η βιβλιοθήκη ECC-LIB . . . 81
6.2.1 GNUMP (GNU Multiple Precission): Αριθμητική Χωρίς Όρια . . . 81
6.2.2 Θέματα Σχεδιασμού της Βιβλιοθήκης ECC-LIB . . . 83
6.2.3 Πειραματικά Αποτελέσματα . . . 85
6.3 Η Εργασία του David J.Malan . . . 86
6.3.1 Κρυπτογραφία Ελλειπτικών Καμπυλών στο $F_{2^p}$ . . . 87
6.3.2 Πρώτη Υλοποίηση EccM 1.0 . . . 88
6.3.3 Δεύτερη Υλοποίηση EccM 2.0 . . . 90
6.4 Η Εργασία TinyECC . . . 91
6.4.1 Αρχές Σχεδίασης του TinyECC . . . 91
6.4.2 Τεχνικές Βελτιστοποίσης του TinyECC . . . 92
6.4.3 Απόδοση του TinyECC . . . 94
6.5 Περιγραφή του Πρωτοκόλλου μας . . . 98
6.5.1 Φόρτωση της Ελλειπτικής Καμπύλης και του Σημείου Βάσης στους Κόμβους . . . 99
6.5.2 Δημιουργία Ιδιωτικού Κλειδιού . . . 100
6.5.3 Πρόσθεση, Βαθμωτός Πολλαπλασιασμός Σημείων Ελλειπτικής Καμπύλης και Δημιουργία Δημοσίου Κλειδιού 100
6.5.4 Elliptic Curve Diffie-Hellman . . . 100
6.5.5 Μηχανισμοί Κρυπτογράφησης και Αποκρυπτογράφησης Δεδομένων . . . 101
6.6 Απόδοση του Πρωτοκόλλου και Συμπεράσματα . . . 105
6.6.1 Απόδοση του Πρωτοκόλλου . . . 105
6.6.2 Σύγκριση του Πρωτοκόλλου μας με Προηγούμενες Εργασίες . . . 106
6.6.3 Συμπεράσματα . . . 108

**7 Πηγαίος Κώδικας 111**
7.1 Εισαγωγή . . . 111
7.2 Το αρχείο "structs.h" . . . 111

7.3 Το αρχείο "ecc.h" . . . . . 114
7.4 Τα αρχεία "sha.h" και "sha1.h" . . . . . 137
7.5 Το αρχείο "TimeProviderDemoApplication.cpp" . . . . . 147

# Κατάλογος Σχημάτων


2.1 Θεμελιώδη στοιχεία της κρυπτογραφίας. . . . . 15
2.2 Επικοινωνία δύο μελών χρησιμοποιώντας κρυπτογράφηση. . . . . 17
2.3 Επικοινωνία δύο μελών χρησιμοποιώντας συμμετρική κρυπτογράφηση.Το κλειδί αποκρυπτογράφησης $d$ μπορεί να υπολογιστεί εύκολα από το κλειδί κρυπτογράφησης $e$. . . . . 20
2.4 Κρυπτογράφηση με τεχνικές κρυπτογραφίας δημοσίου κλειδιού. . 22
2.5 Σχηματική απεικόνιση της κρυπτογραφίας δημοσίου κλειδιού. . . 23
2.6 Μία επίθεση προσωποποίησης στην επικοινωνία δύο οντοτήτων. 25
2.7 Η επίθεση τύπου man in the middle. . . . . 36
2.8 Σχήμα ψηφιακής υπογραφής με ανάκτηση μηνύματος. . . . . 39

3.1 Η ελλειπτική καμπύλη με εξίσωση $y^2 = x^3 - 4x + 0.67$. . . . . 42
3.2 Πρόσθεση δύο σημείων $P, Q$ μιας ελλειπτικής καμπύλης. . . . . 43
3.3 Διπλασιασμός του σημείου $P$ μιας ελλειπτικής καμπύλης. . . . . 44
3.4 Τα σημεία της ελλειπτικής καμπύλης με εξίσωση $y^2 = x^3 + x$ πάνω από το πεδίο $F_{23}$. . . . . 46
3.5 Τα σημεία της εξίσωσης $y^2+xy = x^3+g^4x^2+1$ ορισμένης πάνω από το δυαδικό πεδίο $F_{2^4}$. . . . . 49
3.6 Το πρωτόκολλο Elliptic Curve Diffie Hellman. . . . . 53

4.1 Ένα τυπικό δίκτυο αισθητήρων. . . . . 59
4.2 Απειλές Ενάντια σε Ασύρματα Δίκτυα Αισθητήρων . . . . . 60
4.3 Η επίθεση τύπου wormhole σε ένα δίκτυο αισθητήρων. . . . . 66
4.4 Η επίθεση τύπου HELLO flood σε ένα δίκτυο αισθητήρων. . . . 67

5.1 ΄ iSense. . . . . 72
5.2 Τα στοιχεία υλικού της πλατφόρμας iSense. . . . . 73
5.3 iSense Core Module. . . . . 74
5.4 Το power module 1/2AA Battery της πλατφόρμας iSense. . . . 75
5.5 Τα power modules της πλατφόρμας iSense. . . . . 75
5.6 iSense Gateway Module με USB καλώδιο. . . . . 76
5.7 Η δομή του λογισμικού της πλατφόρμας iSense. . . . . 76

6.1 Η αρχιτεκτονική της βιβλιοθήκης ECC-LIB. . . . . 84
6.2 Χρόνοι δημιουργίας κλειδιού στο EccM 1.0.Για κλειδιά μήκους 63-bit το πρωτόκολλο δεν απέφερε αποτελέσματα. . . . . 89
6.3 Κατανάλωση μνήμης στο EccM 1.0. Κλειδιά μήκους 63-bit εξαντλούν τη RAM των συσκευών MICA 2. . . . . 89
6.4 Χρόνοι εκτέλεσης των λειτουργιών του TinyECC στις διάφορες πλατφόρμες όταν όλες οι τεχνικές βελτιστοποίησης είναι ενεργοποιημένες. . . . . 95
6.5 Κατανάλωση μνήμης RAM του TinyECC όταν όλες οι τεχνικές βελτιστοποίησης είναι ενεργοποιημένες. . . . . 96
6.6 Κατανάλωση μνήμης ROM του TinyECC όταν όλες οι τεχνικές βελτιστοποίησης είναι ενεργοποιημένες. . . . . 96
6.7 Χρόνοι εκτέλεσης των λειτουργιών του TinyECC στις διάφορες πλατφόρμες όταν όλες οι τεχνικές βελτιστοποίησης είναι απενεργοποιημένες. . . . . 97
6.8 Κατανάλωση μνήμης RAM του TinyECC όταν όλες οι τεχνικές βελτιστοποίησης είναι ενεργοποιημένες. . . . . 98
6.9 Κατανάλωση μνήμης ROM του TinyECC όταν όλες οι τεχνικές βελτιστοποίησης είναι ενεργοποιημένες. . . . . 98
6.10 Σύγκριση των χρόνων εκτέλεσης των λειτουργιών Encrypt, Decrypt και ECDH ανάμεσα στο πρωτοκολλό μας στη πλατφόρμα iSense και το TinyECC στις διάφορες πλατφόρμες όταν όλες οι τεχνικές βελτιστοποίησης είναι ενεργοποιημένες. . . . . 108
6.11 ´ ´ ´ ´ Encrypt, Decrypt και ECDH ανάμεσα στο πρωτοκολλό μας στη πλατφόρμα iSense και το TinyECC στις διάφορες πλατφόρμες όταν όλες οι τεχνικές βελτιστοποίησης είναι απενεργοποιημένες. . . . . 109

# Κατάλογος Πινάκων

3.1 Μέγεθος κλειδιών σε bits για ισοδύναμο επίπεδο ασφάλειας. . . 56

4.1 Χαρακτηριστικά των πρωτοκόλλων ασφάλειας . . . . . 70

6.1 Επεξεργαστικός χρόνος σε msec των διαφόρων στοιχείων της βιβλιοθήκης ECC-LIB. . . . . 85
6.2 Σύγκριση της κατανάλωσης μνήμης ανάμεσα στις υλοποιήσεις EccM-1.0 και EccM-2.0. . . . . 91
6.3 Απόδοση της υλοποίησης EccM-2.0. . . . . 91
6.4 Χρόνος εκτέλεσης σε sec των διάφορων λειτουργιών στις συσκευές iSense. . . . . 105
6.5 Κατανάλωση μνήμης της υλοποιησής μας στη συσκευή iSense. . 106
6.6 Σύγκριση των λειτουργιών γέννησης ιδιωτικού και δημοσίου κλειδιού ανάμεσα στις υλοποιήσεις EccM-2.0 του Malan στις συσκευές MICA2 και της δικιάς μας στις συσκευές iSense. . . . 107

# Κεφάλαιο 1

# Εισαγωγή

## 1.1 Ασύρματα Δίκτυα Αισθητήρων

Οι ασύρματες κινητές επικοινωνίες άλλαξαν δραστικά την καθημερινότητα του μέσου πολίτη, αποτέλεσαν παγκοσμίως την ατμομηχανή της ανάπτυξης και της αγοράς εργασίας τις τελευταίες δεκαετίες και έθεσαν ανεξίτηλα τη σφραγίδα τους στην ταυτότητα της σύγχρονης εποχής.Έννοιες όπως «ευρυζωνικότητα», «τηλεργασία/τηλεδιάσκεψη», «εικονική πραγματικότητα», 3G, GPRS, WiFi, κτλ. ενσωματώθηκαν στο λεξιλόγιο των μη ειδικών, χωρίς πολλές φορές να είναι πλήρως κατανοητή η σημασία τους.Η ποιοτική, αδιάλειπτη διασύνδεση και επικοινωνία, οποτεδήποτε και οπουδήποτε, αποτελεί πλέον αδιαπραγμάτευτη απαίτηση κάθε συνδρομητή εφοδιασμένου με κινητές τερματικές συσκευές.

Ως άμεση συνέπεια αυτής της κατάστασης μια νέα πραγματικότητα διαμορφώνεται σήμερα μέσα από την ανάπτυξη των Ασύρματων Δικτύων Αισθητήρων τα οποία είτε λειτουργούν αυτοτελώς, είτε είναι διασυνδεδεμένα στα μεγαλύτερα δίκτυα τηλεπικοινωνιών ή στο διαδίκτυο.Τα δίκτυα αυτά αποτελούνται από μεγάλο αριθμό μικρών ηλεκτρονικών διατάξεων (αισθητήρων), κινητών ή μη, που αποστέλλουν σε μια κεντρική μονάδα πληθώρα δεδομένων προς επεξεργασία και λήψη αποφάσεων.

Η ταχύτατη ανάπτυξη της μικροηλεκτρονικής και των υλικών επέτρεψε την κατασκευή πολύ μικρών αισθητήρων, οι οποίοι έχουν την ικανότητα να μετρούν και να καταγράφουν μια κυριολεκτικά ατέλειωτη σειρά από περιβαλλοντολογικά ή βιολογικά μεγέθη, όπως τη θερμοκρασία, την ατμοσφαιρική πίεση, την υγρασία, τη φωτεινότητα, τη στάθμη υδάτων, την ωρίμανση καρπών, την ανίχνευση χημικών στοιχείων, την πίεση αίματος, τους σφυγμούς καρδιάς, την κίνηση αντικειμένων και ανθρώπων και πολλές ακόμα παραμέτρους που προστίθενται διαρκώς στον παραπάνω κατάλογο.Αξιοσημείωτο είναι ότι σε μία διάταξη ίση με ένα νόμισμα 2 ευρώ μπορούμε να συμπεριλάβουμε πολλά από τα παραπάνω

αισθητήρια και να καταμετρούμε συγχρόνως διάφορα μεγέθη.

Παράλληλα, ανάλογη πρόοδος συντελέστηκε και στη σχεδίαση και υλοποίηση ειδικών πομποδεκτών που επιτρέπουν την αποτελεσματική διασύνδεση των διατάξεων μεταξύ τους και με την κεντρική μονάδα με τεχνολογίες ασύρματης δικτύωσης, αξιολογημένες στα παγκόσμια δίκτυα κινητών επικοινωνιών.Το χαμηλό κόστος παραγωγής παρέχει τη δυνατότητα εγκατάστασης πολύ μεγάλων δικτύων με εκατοντάδες ή χιλιάδες στοιχεία με προηγμένο λογισμικό και ικανότητα να αυτοοργανώνονται, να βελτιστοποιούν και να διασφαλίζουν τη λειτουργία τους χωρίς ιδιαίτερη συντήρηση για μεγάλο χρονικό διάστημα.

Είναι ατέλειωτη η λίστα των εφαρμογών των δικτύων αισθητήρων, ενώ πολλές από αυτές μας είναι ήδη οικείες καθώς ανταποκρίνονται στις συνήθεις δραστηριότητες και ανάγκες μας: μετρήσεις ακριβείας πολλών ατμοσφαιρικών και μετεωρολογικών παραμέτρων, επιτήρηση δασών, υδροβιότοπων, θερμοκηπίων και γενικά αγροτικών καλλιεργειών για έλεγχο υγρασίας, θερμοκρασίας, πίεσης, ωρίμανσης καρπών, επιτήρηση υγρών στοιχείων για ρύπους ή έλεγχο ακραίων φαινομένων όπως οι πλημμύρες, επιτήρηση βιομηχανικού περιβάλλοντος για την εξασφάλιση επιθυμητών συνθηκών της παραγωγικής διαδικασίας, στοιχειώδεις ρυθμίσεις λειτουργιών σε κτίρια όπως θέρμανση, φωτισμός, συναγερμοί.Ως λιγότερο οικείες αλλά αρχαιότερες χρονικά μπορούν να αναφερθούν οι στρατιωτικές εφαρμογές και οι υποβρύχιες εγκαταστάσεις δικτύων για εντοπισμό αντικειμένων τόσο για στρατιωτικές επιχειρήσεις όσο και για αρχαιολογικές έρευνες και πειράματα.Ολες αυτές οι δράσεις αποτελούν κλασικά πλέον -και διόλου ασήμαντα- προϊόντα της επιστημονικής αυτής περιοχής που απαντώνται διεθνώς όχι μόνο σε ανεπτυγμένες αλλά και σε αναπτυσσόμενες χώρες.

Στις πιο πρόσφατες, και επομένως λιγότερο οικείες εφαρμογές μπορούμε να αναφέρουμε τη χρήση δικτύων αισθητήρων για τον εξαιρετικά ακριβή προσδιορισμό της θέσης και της κίνησης αντικειμένων σε εσωτερικούς χώρους, όπως σε κτίρια σε πυκνοδομημένο αστικό περιβάλλον, όπου η απόδοση της κλασικής GPS υπηρεσίας αποδεικνύεται ανεπαρκής.Οι δυνατότητες αυτές θα συμβάλλουν αποφασιστικά στην ασφάλεια και επιτήρηση δημόσιων και ιδιωτικών χώρων.

Στην προσπάθεια για αποτελεσματικότερη διαχείριση του καθημερινού μας περιβάλλοντος πολλές πειραματικές προσπάθειες διεθνώς επικεντρώνονται στην ανάπτυξη και αξιολόγηση δικτύων που δύνανται να εκτελούν φωνητικές εντολές ή να ανιχνεύουν την κίνηση ή την διάθεση των χρηστών τους και να ρυθμίζουν πλήρως εγκαταστάσεις φωτισμού, ηλεκτρικών και ηλεκτροακουστικών συσκευών, ηλεκτρονικής επικοινωνίας, κτλ.

Οι εφαρμογές που αναφέρουμε αλλάζουν επαναστατικά την οργάνωση της κοινωνικής μας ζωής προσφέροντας αναβαθμισμένο περιβάλλον σε χώρους όπου η περίθαλψη ή η διαβίωση ευπαθών ομάδων (υπερήλικες, βρέφη) απαιτεί αδιάκοπτη προσοχή και άμεση επέμβαση.Έτσι, ο σχεδιασμός των «ψηφιακών πόλεων», που αποτελεί το μεγάλο στοίχημα της σύγχρονης πολεοδομίας, θα στηριχθεί

καθοριστικά στις δυνατότητες των ασύρματων δικτύων αισθητήρων.

Ανάλογες εφαρμογές βρίσκονται σε εξέλιξη και στη σχεδίαση και λειτουργία οχημάτων όπου τα ασύρματα δίκτυα αισθητήρων θα μπορούν να λειτουργήσουν συνεργατικά με τον οδηγό για την πλοήγηση του οχήματος, την αποφυγή εμποδίων, την εκκίνηση ή διακοπή της λειτουργίας της μηχανής σε περίπτωση κρίσιμης κατάστασης των επιβαινόντων.Η χρήση των τεχνολογιών αυτών μόνο θετικό αντίκτυπο μπορεί να έχει στην αποφυγή τραγικών γεγονότων και περιστατικών στις οδικές αρτηρίες.

Είναι πράγματι εντυπωσιακές αλλά και ατέρμονες οι προσπάθειες της σύγχρονης επιστημονικής κοινότητας για υποστήριξη της καθημερινότητάς μας με τη χρήση δικτύων αισθητήρων.Τελευταία επικεντρώνεται στη μελέτη των «δικτύων σώματος».Με τα «δίκτυα σώματος» που αποτελούνται από αισθητήρες τοποθετημένους στο ανθρώπινο σώμα ή γύρω από αυτό (ρουχισμό, φόρμες εργατών, αστροναυτών, κτλ.) με σκοπό την καταγραφή των ζωτικών λειτουργιών του (αρτηριακή πίεση, σφυγμούς, ακόμη και ολόκληρα καρδιογραφήματα) είναι δυνατό να συγκεντρώνονται αμέτρητα δεδομένα στον κεντρικό προσωπικό καταγραφέα μας, τα οποία αποτελούν πολύτιμο αρχείο για τον έλεγχο της υγείας και της γενικότερης κατάστασής μας.

Επιπλέον, με τη σύγκλιση των επιστημονικών περιοχών της βιολογίας, των υλικών, της νανοτεχνολογίας και των δικτύων, οι δικτυωμένοι αισθητήρες εντάσσονται στις εσωτερικές λειτουργίες του ανθρώπινου οργανισμού (κυκλοφορία αίματος για συνεχή καταγραφή δεικτών υγείας, όπως πχ. λιπίδια), παρακολουθούν λειτουργίες των βασικών του οργάνων (στομάχι), υποστηρίζουν τα οπτικά νεύρα σε διαδικασίες τεχνικής όρασης, σκιαγραφώντας σταδιακά τον «βιονικό» άνθρωπο του μέλλοντος.

Σήμερα, συνειδητοποιούμε όλο και περισσότερο ότι με τη διασύνδεση και τη διαλειτουργικότητα των ποικίλων ετερόκλητων δικτύων δημιουργείται ένα ισχυρό παγκόσμιο πλαίσιο επικοινωνιών που συντελεί στη δραστική μείωση των αποστάσεων στην παγκόσμια σφαίρα και καταλήγει στην αίσθηση «συγκατοίκησης» στο παγκόσμιο χωριό -ανεξάρτητα από τη χώρα διαβίωσής μας.Τα δίκτυα αισθητήρων προσφέρουν ένα ακόμη επίπεδο στην επικοινωνία αυτή, καλύπτοντας τα κενά όπου υπάρχουν και δημιουργώντας έναν κόσμο «διαχεόμενης» αίσθησης, όπου θα μπορούμε όχι μόνο να ακούμε τη φωνή αλλά και να αισθανόμαστε τους σφυγμούς του μακρινού μας συγκάτοικου.Και αν σκεφθούμε ότι η διασύνδεση των δικτύων αφορά όχι μόνο σε επίγεια δίκτυα, αλλά και σε δορυφορικά κατανοούμε ότι πραγματικά το παγκόσμιο χωριό γίνεται πολύ μικρό.

## 1.2 Ασφάλεια στα Ασύρματα Δίκτυα Αισθητήρων

Από τα παραπάνω, συνειδητοποιούμε ότι τα ασύρματα δίκτυα αισθητήρων βρίσκουν εφαρμογή σε πολλές περιπτώσεις της καθημερινής μας ζωής.Σε πολλές από αυτές, η πληροφορία που διακινείται μέσω αυτών είναι *κρίσιμη*, γεγονός που αυξάνει την ανάγκη για ασφάλεια στα δίκτυα αυτού του τύπου.Η ασύρματη επικοινωνία των συσκευών αισθητήρων και η λειτουργία τους σε όχι και τόσο φιλόξενα περιβάλλοντα τα καθιστά ευάλωτα σε επιθέσεις που στοχεύουν στην υποκλοπή πληροφορίας καθώς και στη διακοπή των βασικών δραστηριοτήτων τους.Η ασφάλεια σε ασύρματα δίκτυα αισθητήρων σχετίζεται με διάφορες προκλήσεις όπως η εμπιστευτικότητα (confidentiality), η ακεραιότητα των δεδομένων (data integrity) και η πιστοποίηση οντοτήτων(entity authentication).Βασικές απαντήσεις στις παραπάνω προκλήσεις παρέχει ο επιστημονικός κλάδος της κρυπτογραφίας.Στα πλαίσια της διπλωματικής αυτής εργασίας, εξετάζεται η εφαρμογή της κρυπτογραφίας και συγκεκριμένα της κρυπτογράφησης ελλειπτικών καμπυλών σε ασύρματα δίκτυα αισθητήρων.Σήμερα, η ασφάλεια στα δίκτυα αισθητήρων αποτελεί ερευνητικό πεδίο το οποίο μελετάται εκτενώς με στόχο την εξασφάλιση της προστασίας των λειτουργιών τους.

## 1.3 Αντικείμενο της Διπλωματικής Εργασίας

Σκοπός της παρούσας διπλωματικής εργασίας είναι η ανάπτυξη ενός πρωτοκόλλου ασφαλούς επικοινωνίας, με χρήση της κρυπτογράφησης ελλειπτικών καμπυλών (ECC: Elliptic Curve Cryptography), σε ασύρματα δίκτυα αισθητήρων.Η ανάπτυξη του πρωτοκόλλου έγινε στην πλατφόρμα iSense η οποία προσφέρει ένα εναρμονισμένο περιβάλλον υλικού και λογισμικού για ανάπτυξη εφαρμογών σε δίκτυα αισθητήρων.Η λειτουργικότητα του πρωτοκόλλου κατά τη διάρκεια ανάπτυξης του, δοκιμάστηκε με τη χρήση του εργαλείου Shawn που αποτελεί έναν εξομοιωτή ασυρμάτων δικτύων αισθητήρων και η απόδοση του εξετάστηκε σε πραγματικές συσκευές της σειράς iSense.Συγκεκριμένα, δημιουργήθηκε ένα μικρό δίκτυο ασυρμάτων συσκευών αισθητήρων iSense, οι οποίες αρχικά εκτελούν τη διαδικασία συμφωνίας κλειδιού Diffie-Hellman με ελλειπτικές καμπύλες, στη συνέχεια ανταλλάσουν μπλοκ δεδομένων τα οποία είναι κρυπτογραφημένα με το κλειδί αυτό και μετά τα αποκρυπτογραφούν.Τέλος, η απόδοση του πρωτοκόλλου συγκρίνεται ως προς το επίπεδο χρονικής καθυστέρησης με προηγούμενες εργασίες.

## 1.4 Διάρθρωση της Διπλωματικής Εργασίας

Η παρούσα διπλωματική εργασία, στα πρώτα κεφάλαια, παρουσιάζει τη βασική θεωρία στην οποία στηρίχτηκε η ανάπτυξη του πρωτοκόλλου μας.

Πιο συγκεκριμένα στο **Κεφάλαιο 2** κάνουμε μια γενική εισαγωγή στην κρυπτογραφία και την ασφάλεια των σύγχρονων συστημάτων.Στο **Κεφάλαιο 3** αναλύουμε τη βασική θεωρία πίσω από τις ελλειπτικές καμπύλες καθώς και τη χρήση τους στη σύγχρονη κρυπτογραφία.Στο **Κεφάλαιο 4**, γίνεται μια περιγραφή των βασικών αρχών ασφάλειας στα ασύρματα δίκτυα αισθητήρων, αναλύονται τα βασικά είδη επιθέσεων καθώς και τα σημαντικότερα πρωτόκολλα ασφαλείας των δικτύων αυτού του τύπου.

Στη συνέχεια, παρουσιάζονται οι τεχνολογίες οι οποίες χρησιμοποιήθηκαν για την ανάπτυξη του πρωτοκόλλου μας.Ειδικότερα, το **Κεφάλαιο 5** αναφέρεται στα βασικά χαρακτηριστικά της πλατφόρμας iSense πάνω στην οποία αναπτύξαμε το πρωτοκολλό μας.

Στο **Κεφάλαιο 6** περιγράφονται οι βασικές λειτουργίες του πρωτοκόλλου μας, αναλύεται η απόδοσή του και συγκρίνεται με προηγούμενες εργασίες.

Τέλος, στο **Κεφάλαιο 7** παρατίθεται ο πηγαίος κώδικας της υλοποίησής μας κατάλληλα σχολιασμένος.

# Κεφάλαιο 2

# Σύγχρονη Κρυπτογραφία και Ασφάλεια Συστημάτων

## 2.1 Εισαγωγή

Η κρυπτογραφία έχει βαθιά και πλούσια ιστορία καθώς χρησιμοποιήθηκε για πρώτη φορά από τους Αιγύπτιους πριν 4000 χρόνια.Επιπλέον, έπαιξε πολύ σημαντικό ρόλο κυρίως κατά τη διάρκεια του πρώτου και του δεύτερου παγκοσμίου πολέμου.Οι βασικοί χρήστες της κρυπτογραφίας σχετίζονταν άμεσα με στρατιωτικές εφαρμογές, με διπλωματικές και κυβερνητικές οργανώσεις καθώς η βασική της χρήση ήταν για την προστασία στρατηγικών και μυστικών πληροφοριών.

Η ανάπτυξη των συστημάτων επικοινωνίας και των υπολογιστών τη δεκαετία του 1960 αποτέλεσε στην ανάγκη για προστασία της πλέον ψηφιακής πληροφορίας.Το 1977 δημοσιεύτηκε από την κυβέρνηση των Ηνωμένων Πολιτειών το πρότυπο DES: Data Encryption Standard το οποίο αποτελεί τον πιο διαδεδομένο μηχανισμό κρυπτογράφησης στην ιστορία.Η πιο ηχηρή ανακάλυψη στην ιστορία της κρυπτογραφίας έγινε το 1976 όταν οι Diffie και Hellman εξέδωσαν την εργασία τους 'New Directions in Cryptography'.Με την εργασία αυτή, διαδόθηκε η έννοια της κρυπτογράφησης δημοσίου κλειδιού καθώς και ένας μηχανισμός ανταλλαγής κλειδιών που βασίζεται στην *δυσκολία* του προβλήματος του διακριτού λογαρίθμου.Παρά το γεγονός ότι οι συγγραφείς δεν παρουσίασαν κάποιο σχήμα κρυπτογράφησης δημοσίου κλειδιού η κοινότητα της κρυπτογραφίας έδειξε μεγάλο ενδιαφέρον.Το 1978 οι Rivest, Shamir και Adleman ανακάλυψαν το πρώτο πρακτικό σχήμα κρυπτογράφησης και υπογραφής που έγινε γνωστό με το όνομα RSA.Το σχήμα RSA έχει τις βάσεις του σε ένα άλλο μαθηματικό πρόβλημα, αυτό της παραγοντοποίησης μεγάλων ακεραίων αριθμών.Κατά την δεκαετία του 1980, προτάθηκαν κάποιοι νέοι μηχανισμοί κρυπτογράφησης δημοσίου κλειδιού από τον El Gamal, οι οποίοι επίσης βασίζονται στο πρόβλημα

του διακριτού λογαρίθμου.

Η πιο σημαντική συνεισφορά της κρυπτογραφίας δημοσίου κλειδιού είναι αυτή της ψηφιακής υπογραφής.Το 1991 υιοθετήθηκε το πρώτο διεθνές πρότυπο για ψηφιακές υπογραφές (ISO/IEC 9796) το οποίο βασίζεται στο σχήμα RSA.Το 1994 η αμερικανική κυβέρνηση παρουσίασε το πρότυπο Digital Signature Standard, ένα μηχανισμό που βασίζεται στο σχήμα El Gamal.Η αναζήτηση για νέα σχήματα δημοσίου κλειδιού και η βελτίωση των υπάρχοντων μηχανισμών συνεχίζεται ακόμα και σήμερα.Συνεχώς προτείνονται νέα πρότυπα ασφαλείας και αναπτύσονται διάφορα προιόντα για την αντιμετώπιση των αναγκών της κοινωνίας της πληροφορίας.

## 2.2 Ασφάλεια Πληροφορίας και Κρυπτογραφία

Για να συνειδητοποιήσει κανείς την έννοια της κρυπτογραφίας πρέπει πρώτα να κατανοήσει τα θέματα της ασφάλειας της πληροφορίας.Η ασφάλεια της πληροφορίας μπορεί να εκφραστεί με διάφορους τρόπους, ανάλογα την κατάσταση και τις απαιτήσεις.Ανεξαρτήτως του ποιοι συμμετέχουν σε μια δοσοληψία, συγκεκριμένες προυποθέσεις που σχετίζονται άμεσα με την ασφάλεια της πληροφορίας πρέπει να ικανοποιούνται.Παραδείγματα τέτοιων προυποθέσεων είναι η ακεραιότητα των δεδομένων, η εμπιστευτικότητα, η αυθεντικότητα και η πιστοποίηση οντότητας.

Κατά το πέρασμα των αιώνων, έχουν αναπτυχθεί διάφοροι μηχανισμοί για την αντιμετώπιση της ασφάλειας της πληροφορίας, όταν αυτή μεταφερόταν μέσω φυσικών εγγράφων.Οι στόχοι της ασφάλειας πληροφορίας δε μπορούν να επιτευχθούν μόνο με πρωτόκολλα και αλγορίθμους, αλλά απαιτούνται διαδικαστικές τεχνικές και συμμόρφωση με το νόμο για το επιθυμητό αποτέλεσμα.Για παράδειγμα, η μυστικότητα ενός γράμματος επιτυγχάνεται με τη χρήση ενός κλειστού φακέλου κατά τη μεταφορά του από μια ταχυδρομική υπηρεσία καθώς και τις προβλέψεις του νόμου για άνοιγμα φακέλων από κάποιον που δεν έχει εξουσιοδότηση.

Στη σύγχρονη εποχή, η πληροφορία πλέον αποθηκεύεται σε μαγνητικούς δίσκους και μεταδίδεται μέσω τηλεπικοινωνιακών συστημάτων.Αυτό που έχει αλλάξει δραματικά σε σχέση με τα παλιότερα χρόνια είναι η δυνατότητα αντιγραφής και τροποποίησης της πληροφορίας.Οποιοσδήποτε έχει τη δυνατότητα να φτιάξει χιλιάδες αντίγραφα της πληροφορίας χωρίς να μπορεί να βρεθεί ποια είναι η αυθεντική.Έτσι λοιπόν αυτό που χρειάζεται στην κοινωνία της ψηφιακής πληροφορίας είναι κάποια μέσα τα οποία εξασφαλίζουν ότι η ασφάλεια της πληροφορίας είναι ανεξάρτητη του φυσικού μέσου.

Ένα θεμελιώδες εργαλείο που χρησιμοποιείται στην ασφάλεια της πληροφορίας είναι η υπογραφή και αποτελεί τη βάση για άλλες υπηρεσίες όπως μη-απάρνηση, εξασφάλιση της προέλευσης των δεδομένων και αναγνώριση.Η υπογραφή ενός ατόμου είναι μοναδική και λειτουργεί ως ένας τρόπος αναγνώρισης και επικύρωσης.Στην εποχή της ψηφιακής πληροφορίας το ζήτημα της υπογραφής πρέπει να εξεταστεί πολύ προσεκτικά.Πολλά σχήματα ψηφιακής υπογραφής έχουν αναπτυχθεί για την εξασφάλιση της προέλευσης της πληροφορίας.Η επίτευξη της ασφάλειας της ψηφιακής πληροφορίας σε μια *ηλεκτρονική* κοινωνία απαιτεί τόσο την χρήση τεχνικών μέσων όσο και νομικών.Τα τεχνικά απαραίτητα μέσα παρέχονται από τον κλάδο της κρυπτογραφίας.

## 2.3 Τι είναι η Κρυπτογραφία

Η λέξη κρυπτογραφία προέρχεται από τα συνθετικά ΄κρυπτός΄ + ΄γράφω΄ και είναι ένας επιστημονικός κλάδος που ασχολείται με την μελέτη, την ανάπτυξη και την χρήση τεχνικών κρυπτογράφησης και αποκρυπτογράφησης με σκοπό την απόκρυψη του περιεχομένου των μηνυμάτων.Αποτελεί ένα κλάδο της επιστήμης της κρυπτολογίας, η οποία ασχολείται με την μελέτη της ασφαλούς επικοινωνίας. Ο κύριος στόχος της είναι να παρέχει μηχανισμούς για δύο ή και περισσότερα μέλη ώστε αυτά να επικοινωνούν χωρίς κάποιος άλλος να είναι ικανός να διαβάζει την πληροφορία εκτός από τα μέλη. Η λέξη κρυπτολογία αποτελείται από την ελληνική λέξη ΄κρυπτός΄ και την λέξη ΄λόγος΄ και χωρίζεται σε δύο κλάδους: την Κρυπτογραφία και την Κρυπτανάλυση.

Ιστορικά η κρυπτογραφία χρησιμοποιήθηκε για την κρυπτογράφηση μηνυμάτων δηλαδή μετατροπή της πληροφορίας από μια κανονική κατανοητή μορφή σε έναν γρίφο, που χωρίς την γνώση του κρυφού μετασχηματισμού θα παρέμενε ακατανόητος. Κύριο χαρακτηριστικό των παλαιότερων μορφών κρυπτογράφησης ήταν ότι η επεξεργασία γινόταν πάνω στην γλωσσική δομή. Στις νεότερες μορφές η κρυπτογραφία κάνει χρήση του αριθμητικού ισοδύναμου, η έμφαση έχει μεταφερθεί σε διάφορα πεδία των μαθηματικών, όπως διακριτά μαθηματικά, θεωρία αριθμών, θεωρία πληροφορίας, υπολογιστική πολυπλοκότητα, στατιστική και συνδυαστική ανάλυση.

Η κρυπτογραφία είναι η μελέτη των μαθηματικών τεχνικών που σχετίζονται με τις προυποθέσεις της ασφαλούς πληροφορίας όπως η ακεραιότητα των δεδομένων, η εμπιστευτικότητα, η αυθεντικότητα οντοτήτων και προέλευσης πληροφορίας.Η κρυπτογραφία δεν αποτελεί μόνο τα μέσα για την παροχή ασφαλούς πληροφορίας αλλά είναι ένα σύνολο τεχνικών που στοχεύουν στα παρακάτω κριτήρια:

- **Εμπιστευτικότητα** (Confidentiality): Το κριτήριο αυτό εμποδίζει όσους δεν έχουν εξουσιοδότηση να έρθουν σε επαφή με το περιεχόμενο της πλη-

ροφορίας.Ένας όρος παρόμοιος με την εμπιστευτικότητα είναι η μυστικότητα.Υπάρχουν διάφορες τεχνικές για την παροχή εμπιστευτικότητας από φυσική προστασία μέχρι μαθηματικούς αλγόριθμους που καθιστούν τα δεδομένα ακατάληπτα.

- **Ακεραιότητα Δεδομένων** (Data Integrity): Το κριτήριο αυτό απαγορεύει την μη-εξουσιοδοτημένη τροποποίηση των δεδομένων.Για την εξασφάλιση της ακεραιότητας των δεδομένων πρέπει να εντοπίζονται με κατάλληλο τρόπο όλες οι μη-εξουσιοδοτημένες παραποιήσεις της πληροφορίας όπως καταχρηστική διαγραφή ή εισαγωγή.Μια κρυπτογραφική μέθοδος κατάλληλη για την εξασφάλιση της ακεραιότητας των δεδομένων είναι η ψηφιακή υπογραφή.

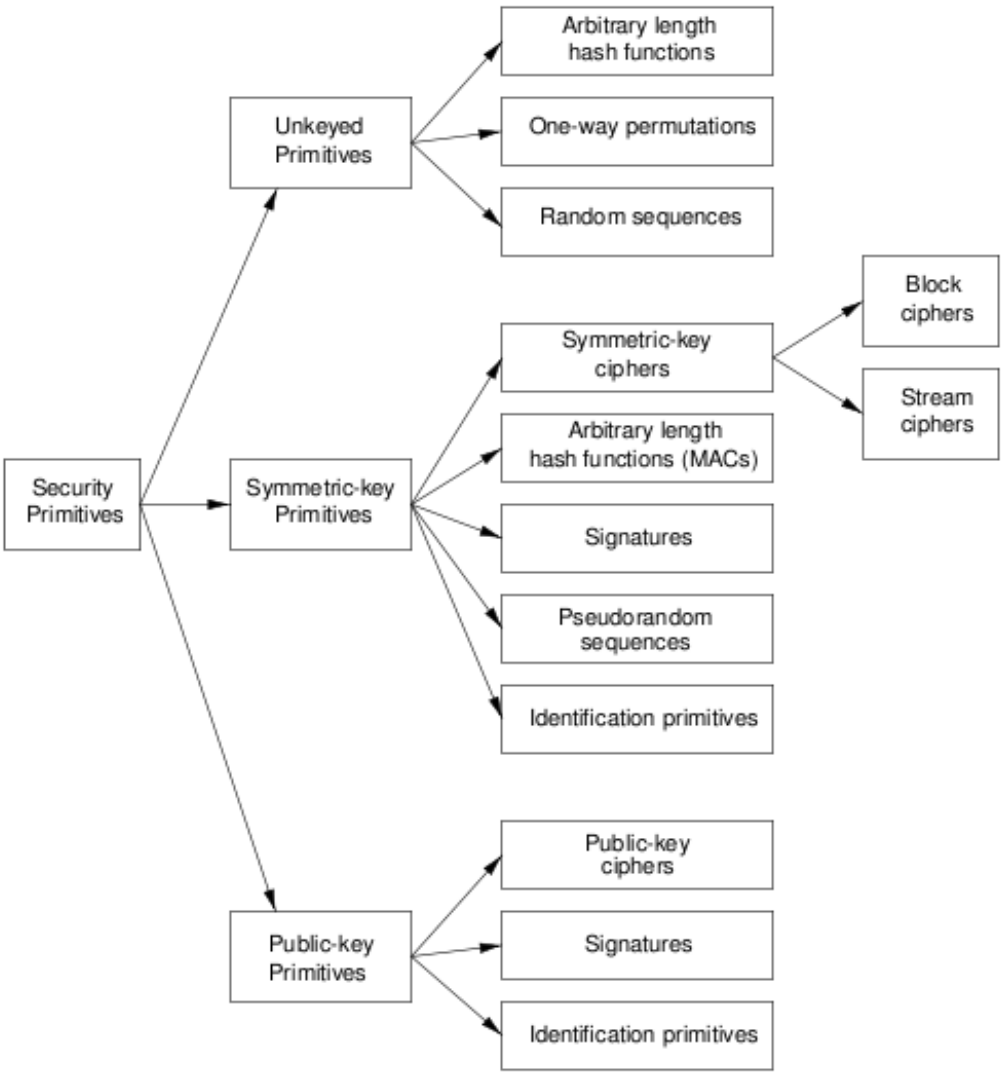
- **Αυθεντικότητα** (Authentication): Η αυθεντικότητα είναι ενα κριτήριο που σχετίζεται με την αναγνώριση και την πιστοποίηση.Αφορά τόσο στην πιστοποίηση μιας οντότητας όσο και στην πιστοποίηση αυθεντικότητας δεδομένων.Στην πρώτη περίπτωση κάθε οντότητα που λαμβάνει μέρος στην επικοινωνία πρέπει να αναγνωρίζεται από τις άλλες.Στη δεύτερη, η αποστολή των δεδομένων θα πρέπει να πιστοποιείται με βάση το περιεχόμενο του, το χρόνο αποστολής κτλ.Η αυθεντικότητα προέλευσης των δεδομένων σιωπηρά εξασφαλίζει και την ακεραιότητα των δεδομένων.

- **Μη Αποποίηση** (Non-repudiation): Το κριτήριο αυτό εμποδίζει μια οντότητα να απαρνηθεί προηγούμενες δεσμεύσεις ή πράξεις.Είναι η μέθοδος για την επίλυση διαφωνιών μεταξύ των οντοτήτων.Για παράδειγμα, εαν μια οντότητα εγκρίνει την εκτέλεση μιας εντολής σε μια άλλη και στη συνέχεια αρνείται την έγκριση αυτή, θα πρέπει να υπάρχει ένας τρόπος ξεκαθαρισμού του τι έγινε.Συνήθως, μια τρίτη έμπιστη οντότητα χρησιμοποιείται για την επίλυση των διαφωνιών.

Η αυθεντικότητα αποτελεί την βάση κάθε ασφαλούς επικοινωνίας και χωρίς αυτή όλα τα παραπάνω κριτήρια δεν είναι τόσο καθοριστικά.Ο θεμελιώδης στόχος της κρυπτογραφίας είναι να ικανοποιεί και τα τέσσερα παραπάνω κριτήρια τόσο στη θεωρία όσο και στην πράξη.Επιπλέον, πρέπει να εμποδίζει και να εντοπίζει τις μη εξουσιοδοτημένες και κακόβουλες ενέργειες.

Στο παρακάτω σχήμα παρουσιάζονται τα θεμελιακά στοιχεία της κρυπτογραφίας, όπως μηχανισμοί κρυπτογράφησης, συναρτήσεις κατακερματισμού και σχήματα ψηφιακής υπογραφής.Τα στοιχεία αυτά πρέπει να αξολογούνται με βάση τις εξής ιδιότητες: επίπεδο ασφάλειας που προσφέρουν, λειτουργικότητα, μεθόδους λειτουργίας, απόδοση (π.χ. αριθμός bits ανά δευτερόλεπτο που μπορεί να κρυπτογραφεί ένας μηχανισμός κρυπτογράφησης) και ευκολία υλοποίησης.Όπως πάντα, η σημασία των παραπάνω ιδιοτήτων εξαρτάται από την

εφαρμογή και τους διαθέσιμους πόρους.Για παράδειγμα, σε ένα περιβάλλον με περιορισμένη υπολογιστική ισχύ μπορεί να χρειαστεί μείωση του επιπέδου ασφαλείας ώστε η γενικότερη απόδοση του συστήματος να είναι καλύτερη.




Σχήμα 2.1: Θεμελιώδη στοιχεία της κρυπτογραφίας.

### 2.3.1 Βασική Ορολογία

Στην ενότητα αυτή παρουσιάζονται βασικές ορολογίες και έννοιες που χρησιμοποιούνται συχνά στην κρυπτογραφία.

### Τομέας Κρυπτογράφησης

- Το σύμβολο A αποτελεί ένα πεπερασμένο σύνολο που ονομάζεται αλφάβητο.Για παράδειγμα, το αλφάβητο $A = \{0, 1\}$ (δυαδικό αλφάβητο) χρησιμοποιείται αρκετά συχνά.Επισημαίνουμε ότι οποιοδήποτε αλφάβητο μπορεί να κωδικοποιηθεί με όρους του δυαδικού αλφαβήτου.

- Το σύμβολο M αποτελεί ένα σύνολο που ονομάζεται *χώρος μηνύματος*(message space).Το M αποτελείται από αλφαριθμητικά συμβόλων του αλφαβήτου ορισμού.Κάθε στοιχείο του M ονομάζεται plaintext message (ακρυπτογράφητο κείμενο) ή απλά plaintext.Παραδείγματος χάριν, το M μπορεί να αποτελείται από δυαδικά αλφαριθμητικά, κείμενο σε κάποια γλώσσα, κώδικα ενός προγράμματος κτλ.

- Το σύμβολο C αποτελεί ένα σύνολο που ονομάζεται ciphertext space(χώρος κρυπτογραφημάτων).Το C αποτελείται από σύμβολα ενός αλφαβήτου ορισμού το οποίο συνήθως διαφέρει από το αλφάβητο M.Κάθε στοιχείο του C ονομάζεται ciphertext( κρυπτογράφημα).

### Μετασχηματισμοί Κρυπτογράφησης/Αποκρυπτογράφησης

- Το σύμβολο K ορίζει ένα σύνολο που ονομάζεται χώρος κλειδιών (key space).Κάθε στοιχείο του K ονομάζεται κλειδί.

- Κάθε στοιχείο $e \in K$ καθορίζει μοναδικά μια 1-1 αντιστοιχία από το M στο C, η οποία συμβολίζεται με $E_e$.Η $E_e$ ονομάζεται συνάρτηση κρυπτογράφησης ή κρυπτογραφικός μετασχηματισμός.

- Για κάθε στοιχείο $d \in K$, το σύμβολο $D_d$ καθορίζει μια 1-1 αντιστοιχία από το C στο M.Η $D_d$ ονομάζεται συνάρτηση αποκρυπτογράφησης ή μετασχηματισμός αποκρυοπτογράφησης.

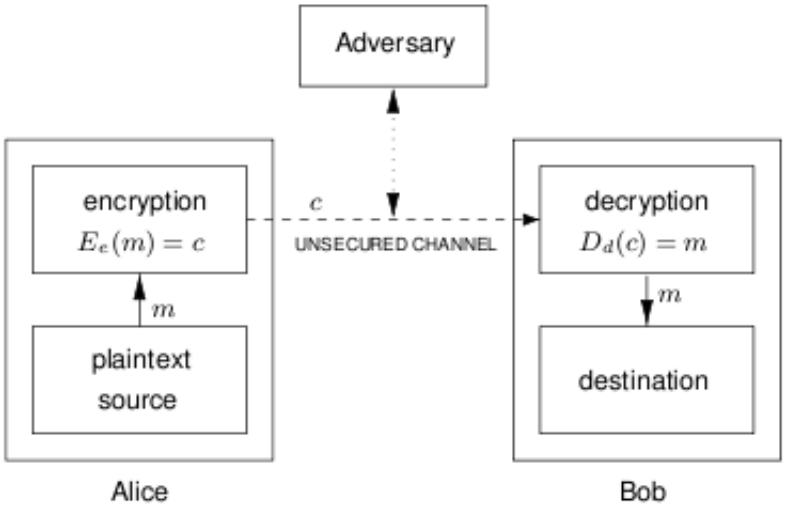
- Η διαδικασία εφαρμογής του μετασχηματισμού $E_e$ πάνω σε ένα μήνυμα $m \in M$ ονομάζεται κρυπτογράφηση του $m$.

- Η διαδικασία εφαρμογής του μετασχηματισμού $D_d$ σε ένα κρυπτογράφημα $c \in C$ ονομάζεται αποκρυπτογράφηση του $c$.

- Ένα σχήμα κρυπτογράφησης αποτελείται από ένα σύνολο μετασχηματισμών κρυπτογράφησης $\{E_e : e \in K\}$ και από ένα σύνολο μετασχηματισμών αποκρυπτογράφησης $\{D_d : d \in K\}$ με την ιδιότητα ότι για κάθε $e \in K$ υπάρχει ένα μοναδικό κλειδί $d \in K$ έτσι ώστε $D_d = E_e^{-1}$.Έτσι, ισχύει ότι $D_d(E_e(m)) = m$ για κάθε $m \in M$.

- Τα κλειδιά $e$ και $d$ συνήθως αναφέρονται ώς ζευγάρι κλειδιών και συμβολίζονται ως $(e, d)$.Αναφέρουμε ότι τα $e$ και $d$ μπορούν να είναι ίδια.

- Για την κατασκευή ενός σχήματος κρυπτογράφησης απαιτείται η επιλογή ενός χώρου μηνύματος $M$, ενός χώρου κρυπτογραφήματος $C$, ενός χώρου κλειδιού $K$, ένα σύνολο μετασχηματισμών κρυπτογράφησης $\{E_e : e \in K\}$ και ένα σύνολο μετασχηματισμών αποκρυπτογράφησης $\{D_d : d \in K\}$.




Σχήμα 2.2: Επικοινωνία δύο μελών χρησιμοποιώντας κρυπτογράφηση.

### Μέλη Επικοινωνίας

- Ένα *μέλος* ή μια *οντότητα* είναι κάποιος ο οποίος αποστέλει, λαμβάνει ή χειρίζεται πληροφορία.Μια οντότητα μπορεί να είναι ένας άνθρωπος, ένας υπολογιστής κτλ.

- Ο *αποστολέας* είναι μια οντότητα της επικοινωνίας που αποστέλει νόμιμη πληροφορία.

- Ο *παραλήπτης* είναι μια οντότητα της επικοινωνίας που είναι ο νόμιμος αποδέκτης της πληροφορίας.

- Ο *επιτιθέμενος* είναι μια οντότητα της επικοινωνίας που δεν είναι ούτε ο αποστολέας ούτε ο παραλήπτης.Προσπαθεί να υπερνικήσει την υπηρεσία

ασφάλειας της πληροφορίας που παρέχεται ανάμεσα στον αποστολέα και τον παραλήπτη.Συνώνυμες έννοιες είναι οι εξής: αντίπαλος, εχθρός και ωτακουστής.Συχνά, ένας επιτιθέμενος προσπαθεί να παίξει τον ρόλο ενός έντιμου μέλους.

### Κανάλια Επικοινωνίας

- Ένα κανάλι είναι το μέσο μετάδοσης της πληροφορίας από μία οντότητα σε μια άλλη.
- Ένα *φυσικά ασφαλές* κανάλι είναι αυτό στο οποίο ένας επιτιθέμενος δεν έχει φυσική πρόσβαση.
- Ένα ανασφαλές κανάλι είναι αυτό στο οποίο μη-εξουσιοδοτημένες οντότητες μπορούν να διαβάσουν, να τροποποιήσουν και να εισάγουν ή να διαγράψουν την πληροφορία που μεταδίδεται.
- Ένα ασφαλές κανάλι είναι αυτό στο οποίο ένας επιτιθέμενος δεν έχει τη δυνατότητα ανάγνωσης, εισαγωγής ή διαγραφής της πληροφορίας που μεταδίδεται.

### Ασφάλεια

Μια θεμελιώδης υπόθεση στην κρυπτογραφία είναι ότι τα σύνολα $M, C, K, \{E_e : e \in K\}, \{D_d : d \in K\}$ αποτελούν κοινή γνώση.Όταν δύο οντότητες επιθυμούν να επικοινωνήσουν με ασφάλεια χρησιμοποιώντας ένα σχήμα κρυπτογράφησης, το μόνο μυστικό που διατηρούν είναι το συγκεκριμένο ζευγάρι κλειδιών $(e, d)$ που έχουν επιλέξει.Για την επίτευξη επιπλέον ασφαλείας κάποιος θα μπορούσε να διατηρήσει μυστικούς και τους μετασχηματισμούς κρυπτογράφησης και αποκρυπτογράφησης αλλά η ιστορία έχει δείξει ότι κάτι τέτοιο είναι αρκετά δύσκολο.

Ένα σχήμα κρυπτογράφησης θεωρείται *εύθραυστο*(breakable), εάν ένα τρίτο μέλος χωρίς γνώση του ζευγαριού κλειδιών $(e, d)$, μπορεί να ανακτήσει το ακρυπτογράφητο κείμενο από το κρυπτογράφημα σε κατάλληλο χρονικό διάστημα.Το κατάλληλο χρονικό διάστημα ορίζεται σε σχέση με τη διάρκεια χρησιμότητας και ζωής της πληροφορίας που προστατεύεται.Ένα σχήμα κρυπτογράφησης μπορεί να *σπάσει* δοκιμάζοντας όλα τα πιθανά κλειδιά του χώρου κλειδιών(εφόσον αυτός είναι κοινής γνώσης).Η μέθοδος αυτή ονομάζεται εξαντλητική αναζήτηση του χώρου κλειδιών.Από το παραπάνω γεγονός, συνεπάγεται ότι ο χώρος κλειδιών πρέπει να έχει τέτοιο μέγεθος ώστε μια επίθεση αυτού του είδους να είναι υπολογιστικά αδύνατη και κάθε δημιουργός σχημάτων κρυπτογράφησης πρέπει να το λαμβάνει υπόψιν του.

Η ασφάλεια της πληροφορίας αποτελεί μια ευρύτερη έννοια η οποία περιλαμβάνει στοιχεία όπως ακεραιότητα δεδομένων και αυθεντικότητα.Μια υπηρεσία ασφάλειας της πληροφορίας είναι μια μέθοδος για την παροχή συγκεκριμένου τύπου ασφαλείας.Το να *σπάσει* κανείς μια τέτοια υπηρεσία σημαίνει να υπερνικήσει τους στόχους της.Στο σημείο αυτό αναφέρουμε ότι υπάρχουν δυο βασικοί τύποι επιτιθέμενων: ο παθητικός και ο ενεργός.Ο παθητικός αντίπαλος έχει την ικανότητα να διαβάζει μόνο την πληροφορία που μεταδίδεται μέσω ενός καναλιού σε αντίθεση με τον ενεργό ο οποίος μπορεί να την τροποποιήσει, να τη διαγράψει και να τη επαναμεταδόσει.

**Κρυπτολογία**

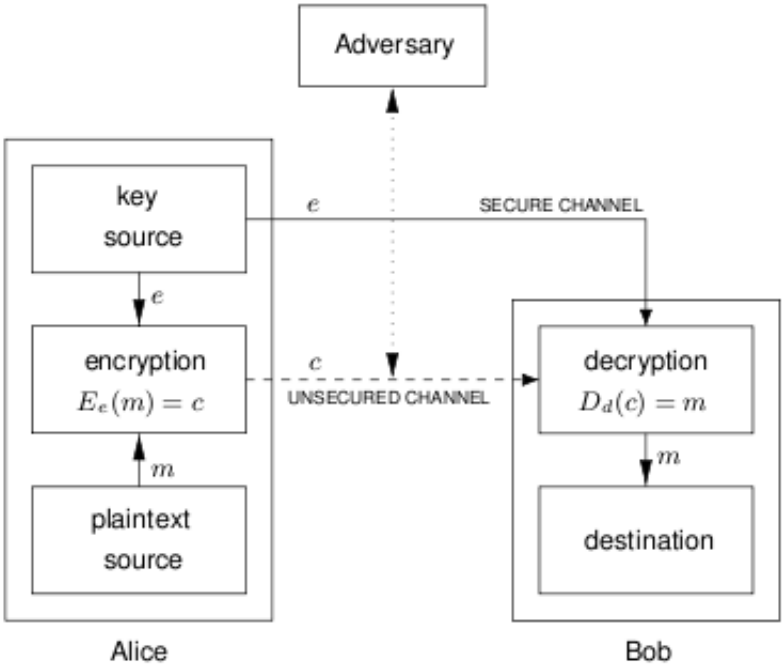
- Κρυπτανάλυση είναι η μελέτη μαθηματικών τεχνικών που χρησιμοποιούνται για να υπερνικήσουν κρυπτογραφικούς μηχανισμούς και γενικότερα υπηρεσίες παροχής ασφάλειας πληροφορίας.
- Κρυπτολογία είναι η μελέτη της κρυπτογραφίας και της κρυπτανάλυσης.
- Κρυπτοσύστημα είναι ένα σύνολο από κρυπτογραφικούς μηχανισμούς που χρησιμοποιούνται για την παροχή υπηρεσιών ασφαλείας της πληροφορίας.

Οι κρυπτογραφικές τεχνικές χωρίζονται σε δύο μεγάλες κατηγορίες: την κρυπτογράφηση συμμετρικού κλειδιού και την κρυπτογράφηση δημοσίου κλειδιού.Οι κατηγορίες αυτές περιγράφονται παρακάτω.

## 2.4 Κρυπτογράφηση Συμμετρικού Κλειδιού

Ας θεωρήσουμε ένα σχήμα κρυπτογράφησης το οποίο αποτελείται από τα σύνολα των μετασχηματισμών κρυπτογράφησης και αποκρυπτογράφησης $\{E_e : e \in K\}$ και $\{D_d : d \in K\}$ αντίστοιχα, όπου $K$ είναι ο χώρος κλειδιού(key space).Το σχήμα κρυπτογράφησης λέγεται *συμμετρικού κλειδιού* εαν για κάθε ζευγάρι κλειδιών $(e, d)$ είναι υπολογιστικά εύκολο να βρεθεί το $d$ γνωρίζοντας μόνο το $e$ και αντιστοίχως να βρεθεί το $e$ γνωρίζοντας το $d$.Στα πιο γνωστά κρυπτοσυστήματα συμμετρικού κλειδιού ισχύει ότι $d = e$.Άλλοι όροι που χρησιμοποιούνται είναι οι κρυπτογραφία ενός κλειδιού, κρυπτογραφία ιδιωτικού κλειδιού και συμβατική κρυπτογραφία.Το παρακάτω σχήμα αναπαριστά την επικοινωνία δύο οντοτήτων που χρησιμοποιούν κρυπτογράφηση συμμετρικού κλειδιού.

Ένα μείζων θέμα στα συμμετρικά κρυπτοσυστήματα είναι να βρεθεί μία αποτελεσματική μέθοδος για την ασφαλή ανταλλαγή των κλειδιών μεταξύ των οντοτήτων.Το πρόβλημα αυτό είναι γνωστό ως *πρόβλημα διανομής κλειδιών*.Στην




Σχήμα 2.3: Επικοινωνία δύο μελών χρησιμοποιώντας συμμετρική κρυπτογράφηση.Το κλειδί αποκρυπτογράφησης $d$ μπορεί να υπολογιστεί εύκολα από το κλειδί κρυπτογράφησης $e$.

κρυπτογράφηση συμμετρικού κλειδιού υποτίθεται ότι όλα τα μέλη γνωρίζουν το σύνολο των μετασχηματισμών κρυπτογράφησης/αποκρυπτογράφησης.Όπως αναφέραμε και προηγουμένως η μόνη πληροφορία που παραμένει μυστική είναι το κλειδί $d$ καθώς και το $e$ αφού το $d$ προκύπτει από αυτό.Υπάρχουν δύο βασικές κατηγορίες σχημάτων συμμετρικής κρυπτογράφησης: η block cipher και η stream cipher.

### 2.4.1 Block Cipher

Το block cipher αποτελεί ένα σχήμα κρυπτογράφησης το οποίο κατακερματίζει τα αρχικά μηνύματα και τα μεταδίδει σε μπλοκ δεδομένου μεγέθους $t$ πάνω από ένα αλφάβητο $A$.Το σχήμα αυτό κρυπτογραφεί κάθε μπλοκ ξεχωριστά.Οι πιο γνωστές τεχνικές κρυπτογράφησης συμμετρικού κλειδιού ανήκουν στην κατηγορία αυτή όπως οι αλγόριθμοι DES,FEAl και RC5.Οι βασικές υποκατηγορίες της block cipher είναι οι substitution cipher και transposition cipher.Στην πρώτη υποκατηγορία τα σύμβολα των μπλοκ δεδομένων αντικαθίστανται από

άλλα σύμβολα ή ομάδες συμβόλων ενώ στη δεύτερη γίνεται μετάθεση των συμβόλων των μπλοκ δεδομένων.

### 2.4.2 Stream Cipher

Τα stream cipher αποτελούν μια σημαντική κατηγορία των σχημάτων κρυπτογράφησης συμμετρικού κλειδιού.Υπό μία έννοια, αποτελούν απλά block cipher με μέγεθος μπλοκ ίσο με ένα.Η ιδιότητα που τα κάνει ιδιαίτερα εύχρηστα είναι ότι ο μετασχηματισμός κρυπτογράφησης μπορεί να αλλάζει για κάθε διαφορετικό σύμβολο πληροφορίας που κρυπτογραφείται.Σε περιπτώσεις που τα λάθη μετάδοσης είναι πολύ συχνά, τα stream cipher έχουν πλεονέκτημα καθώς δεν έχουν σφάλμα διάδοσης.Επίσης μπορούν να χρησιμοποιηθούν αποτελεσματικά όταν τα δεδομένα πρέπει να επεξεργάζονται ένα σύμβολο τη φορά όπως π.χ. σε περιπτώσεις που ο εξοπλισμός δεν διαθέτει επαρκή μνήμη για αποθήκευση δεδομένων.Για τον ορισμό των stream cipher είναι απαραίτητη η έννοια του keystream.Δεδομένου του χώρου κλειδιών $K$ και ενός συνόλου μετασχηματισμών κρυπτογράφησης, μια ακολουθία συμβόλων $e_1e_2...e_i \in K$, ονομάζεται keystream.Ας θεωρήσουμε το $A$ ως ένα αλφάβητο συμβόλων και $E_e$ ένα απλό substitution cipher με μέγεθος μπλοκ ένα, όπου $e \in K$.Η ακολουθία $m_1m_2m_3...$ είναι το ακρυπτογράφητο αλφαριθμητικό και $e_1e_2e_3...$ είναι ένα keystream από το $K$.Ένα stream cipher μετατρέπει το ακρυπτογράφητο αλφαριθμητικό στο κρυπτογράφημα $c_1c_2c_3....$ όπου $c_i = E_{e_i}(m_i)$.Εάν το $d_i$ αναπαριστά τον αντίστροφο του $e_i$ τότε η συνάρτηση $D_{d_i}(c_i) = m_i$ αποκρυπτογραφεί το κρυπτογραφημένο μήνυμα.

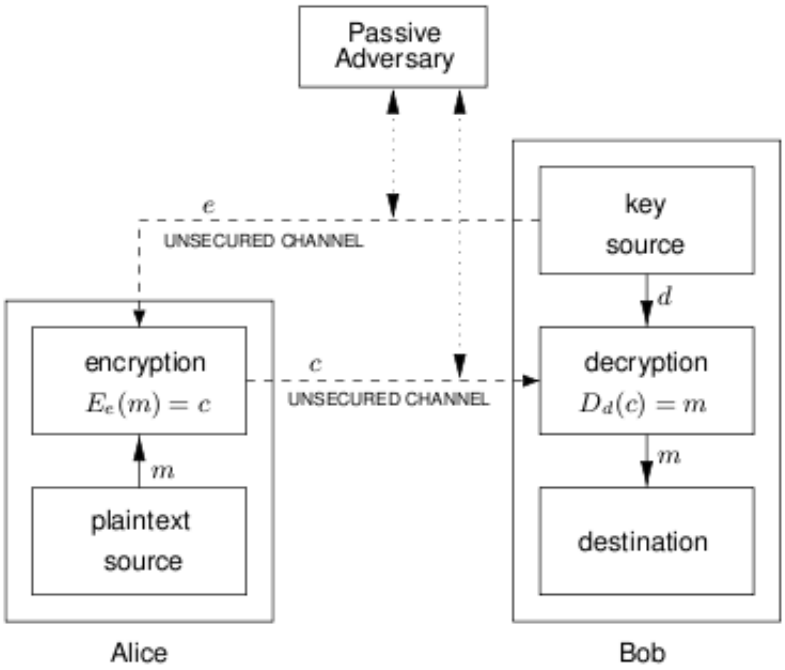
Ένα stream cipher υλοποιεί απλά σχήματα κρυπτογράφησης ανάλογα με το keystream που χρησιμοποιείται.Το keystream μπορεί να δημιουργείται τυχαία, η από έναν αλγόριθμο ο οποίο γεννά νέα keystreams από ένα αρχικό που ονομάζεται seed.

## 2.5 Κρυπτογράφηση Δημοσίου Κλειδιού

Ας θεωρήσουμε ένα σχήμα κρυπτογράφησης το οποίο αποτελείται από τα σύνολα των μετασχηματισμών κρυπτογράφησης και αποκρυπτογράφησης $\{E_e : e \in K\}$ και $\{D_d : d \in K\}$ αντίστοιχα, όπου $K$ είναι ο χώρος κλειδιού(key space).Ας θεωρήσουμε ένα τυχαίο ζευγάρι μετασχηματισμών κρυπτογράφησης/αποκρυπτογράφησης $(E_e, D_d)$ και ας υποθέσουμε ότι για κάθε ζευγάρι είναι αδύνατο, γνωρίζοντας το $E_e$ και δεδομένου ενός κρυπτογραφήματος $c \in C$, να υπολογίσουμε το μήνυμα $m \in M$ ώστε $E_e(m) = c$.Αυτή η ιδιότητα συνεπάγεται ότι δεδομένου του $e$ είναι αδύνατο να καθορίσουμε το κλειδί αποκρυπτογράφησης $d$.Η συνάρτηση $E_e$ θεωρείται σαν μία one-way trapdoor συνάρτηση, με το $d$

να αποτελεί την trapdoor πληροφορία η οποία είναι απαραίτητη για τον υπολογισμό την αντίστροφη συνάρτηση και να επιτραπεί η αποκρυπτογράφηση.Αυτό είναι διαφορετικό σε σχέση με την κρυπτογράφηση συμμετρικού κλειδιού που τα κλειδιά $e$ και $d$ είναι πρακτικά ίδια.

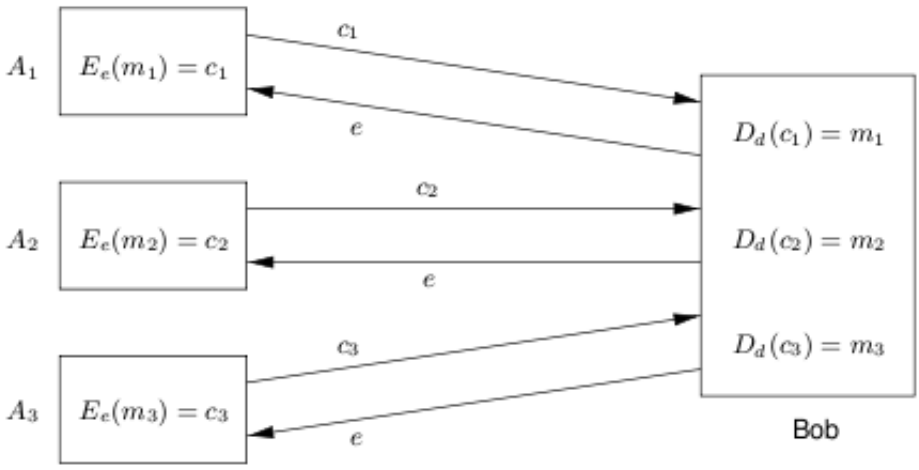
Υπό αυτές τις συνθήκες, ας θεωρήσουμε την επικοινωνία δύο οντοτήτων (Alice-Bob)όπως αυτή φαίνεται στο παρακάτω σχήμα.Ο Bob επιλέγει το ζευγάρι κλειδιών $(e, d)$ και στέλνει στην Alice το κλειδί $e$, που ονομάζεται και δημόσιο κλειδί, μέσω ενός οποιοδήποτε καναλιού.Το κλειδί $d$, που ονομάζεται ιδιωτικό κλειδί, το κρατάει μυστικό και ασφαλές.Η Alice μπορεί να στείλει ένα μήνυμα $m$ στον Bob εφαρμόζοντας το μετασχηματισμό κρυπτογράφησης που καθορίζεται από το δημόσιο κλειδί του Bob, $c = E_e(m)$.Ο Bob αποκρυπτογραφεί το μήνυμα $c$ εφαρμόζοντας τον αντίστροφο μετασχηματισμό $D_d$ που καθορίζεται από το $d$.




Σχήμα 2.4: Κρυπτογράφηση με τεχνικές κρυπτογραφίας δημοσίου κλειδιού.

Παρατηρούμε ότι η παραπάνω εικόνα σε σχέση με εκείνη της συμμετρικής κρυπτογράφησης διαφέρει.Σε αυτήν την περίπτωση το κλειδί κρυπτογράφησης μεταδίδεται πάνω από ένα ανασφαλές κανάλι.Το κανάλι αυτό μπορεί να είναι το ίδιο με αυτό πάνω από το οποίο θα μεταδοθούν τα κρυπτογραφημένα δεδομένα.Από τη στιγμή που το κλειδί κρυπτογράφησης $e$ δε χρειάζεται να παραμένει μυστικό γίνεται κοινώς γνωστό και οποιαδήποτε οντότητα μπορεί να στείλει κρυ-

πτογραφημένα μηνύματα στον Bob, ο οποίος είναι ο μόνος που μπορεί να τα αποκρυπτογραφήσει.Η παρακάτω εικόνα απεικονίζει αυτήν την ιδέα, θεωρώντας τα $A1, A2$ και $A3$ ως διακριτές οντότητες.



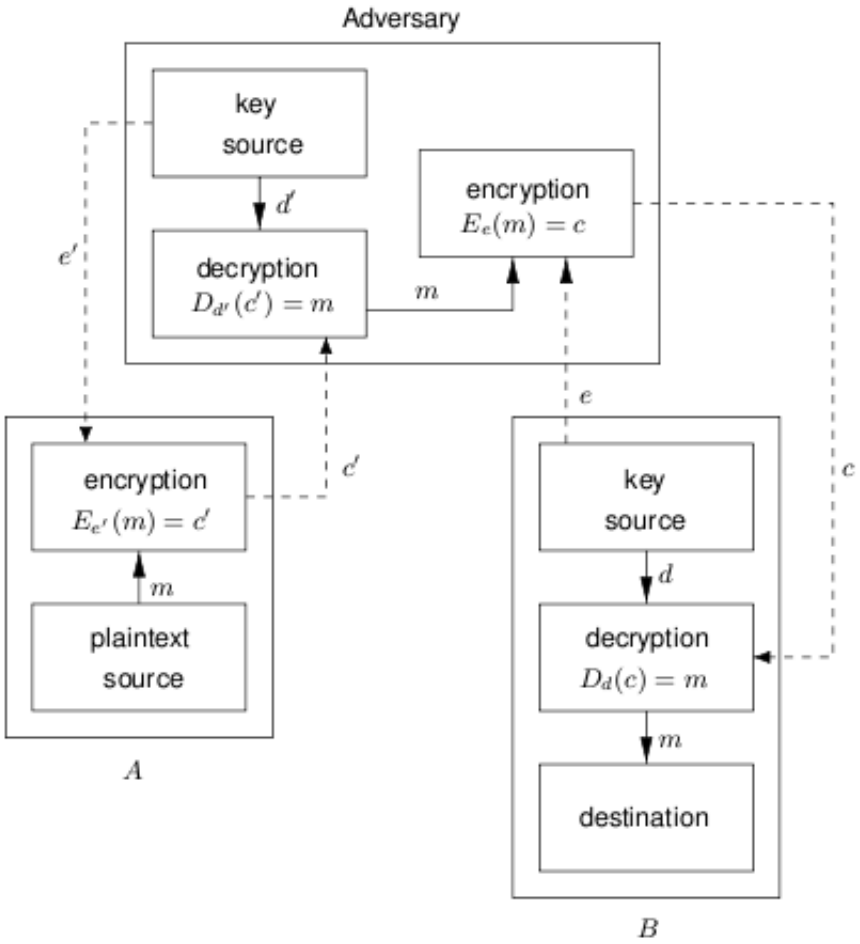

Σχήμα 2.5: Σχηματική απεικόνιση της κρυπτογραφίας δημοσίου κλειδιού.

Σαν ένα φυσικό ανάλογο της κρυπτογραφίας δημοσίου κλειδιού, ας θεωρήσουμε ένα μεταλλικό κουτί του οποίου η κλειδαρία προστατεύεται από κάποιον μυστικό συνδυασμό, τον οποίο μόνο ο Bob γνωρίζει.Αν η κλειδαρία αφεθεί δημοσίως ανοιχτή, οποιοσδήποτε μπορεί να τοποθετήσει ένα μήνυμα μέσα και κατόπν να την κλειδώσει.Μόνο ο Bob μπορεί να λάβει το μήνυμα αφού είναι ο μόνος που ξέρει το μυστικό συνδυασμό της κλειδαριάς και μπορεί να την ανοίξει.Ακόμα και η οντότητα που τοποθέτησε το μήνυμα μέσα στο κουτί δεν μπορεί να το ανακτήσει.Η κρυπτογραφία δημοσίου κλειδιού προυποθέτει τη γνώση ενός δημοσίου κλειδιού $e$ η οποία όμως δεν επιτρέπει τον υπολογισμό του ιδιωτικού κλειδιού $d$.

Ας θεωρήσουμε ένα σχήμα κρυπτογράφησης το οποίο αποτελείται από τα σύνολα των μετασχηματισμών κρυπτογράφησης και αποκρυπτογράφησης $\{E_e : e \in K\}$ και $\{D_d : d \in K\}$ αντίστοιχα, όπου $K$ είναι ο χώρος κλειδιού(key space).Η μέθοδος κρυπτογράφησης θεωρείται σχήμα κρυπτογράφησης δημοσίου κλειδιού εάν για κάθε ζευγάρι $(e, d)$ το κλειδί $e$ γίνεται δημοσίως γνωστό (δημόσιο κλειδί) ενώ το $d$ παραμένει μυστικό (ιδιωτικό κλειδί).Για τη διατήρηση της ασφάλειας αυτού του σχήματος, θα πρέπει να είναι αδύνατο να υπολογιστεί το $d$ από το $e$.

### 2.5.1 Η ανάγκη για αυθεντικότητα στα κρυπτοσυστήματα δημοσίου κλειδιού

Υπό κάποιες συνθήκες, η κρυπτογραφία δημοσίου κλειδιού θα φαινόταν ως ιδανικό κρυπτοσύστημα καθώς δεν απαιτεί ένα ασφαλές κανάλι για τη μετάδοση του κλειδιού κρυπτογράφησης.Αυτό θα συνεπαγόταν ότι δύο οντότητες θα μπορούσαν να επικοινωνήσουν πάνω από ένα ανασφαλές κανάλι χωρίς να ανταλλάξουν κλειδιά.Δυστυχώς, κάτι τέτοιο δεν είναι αλήθεια.Η παρακάτω εικόνα παρουσιάζει πως ένας ενεργός επιτιθέμενος μπορεί να νικήσει το κρυπτοσύστημα χωρίς *σπάσει* το μηχανισμό κρυπτογράφησης.Το σχήμα που παρουσιάζεται είναι ένας τύπος προσωποποίησης(impersonation) και είναι ένα παράδειγμα αποτυχίας του πρωτοκόλλου.Στο σενάριο αυτό ο επιτιθέμενος προσποιείται στον $A$ ότι είναι η οντότητα $B$ στέλνοντας του ένα κλειδί $e'$ το οποίο ο $A$ υποθέτει ότι είναι το δημόσιο κλειδί του $B$.Με τον τρόπο αυτό ο επιτιθέμενος λαμβάνει τα μηνύματα που θέλει ο $A$ να στείλει στο $B$, κρυπτογραφημένα με το δικό του δημόσιο κλειδί.Τα αποκρυπτογραφεί με το ιδιωτικό του κλειδί $d'$ και στη συνέχεια τα κρυπτογραφεί με το δημόσιο κλειδί του $B$ και του τα στέλνει.Το γεγονός αυτό τονίζει την ανάγκη για αυθεντικότητα προέλευσης των δεδομένων στα κρυπτοσυστήματα δημοσίου κλειδιού.Η οντότητα $A$ πρέπει να είναι πεπεισμένη ότι κρυπτογραφεί τα δεδομένα της με το έντιμο δημόσιο κλειδί της οντότητας $B$.




Σχήμα 2.6: Μία επίθεση προσωποποίησης στην επικοινωνία δύο οντοτήτων.

## 2.6 Κρυπτογραφία Συμμετρικού Κλειδιού Εναντίον Κρυπτογραφίας Δημοσίου Κλειδιού

Τα σχήματα κρυπτογράφησης συμμετρικού κλειδιού και δημοσίου κλειδιού παρουσιάζουν διάφορα πλεονεκτήματα και μειονεκτήματα, κάποια από τα οποία είναι κοινά και στα δύο.Παρακάτω αναφέρονται τα βασικά θετικά και αρνητικά των σχημάτων αυτών.

### 2.6.1 Πλεονεκτήματα Κρυπτογραφίας Συμμετρικού Κλειδιού

1. Τα κρυπτοσυστήματα συμμετρικού κλειδιού μπορούν να σχεδιαστούν έτσι ώστε να έχουν υψηλούς ρυθμούς απόδοσης δεδομένων όπως π.χ. ρυθμούς κρυπτογράφησης δεδομένων της τάξης εκατοντάδων Megabytes το δευτερόλεπτο.

2. Τα κλειδιά που χρησιμοποιούνται στη συμμετρική κρυπτογράφηση είναι σχετικά μικρά.

3. Τα κρυπτογραφήματα συμμετρικού κλειδιού μπορούν να αποτελέσουν τη βάση για τη δημιουργία διαφόρων κρυπτογραφικών μηχανισμών όπως γεννήτριες ψευδο-τυχαίων αριθμών, συναρτήσεις κατακερματισμού και σχήματα ψηφιακής υπογραφής(περιγράφονται παρακάτω).

4. Τα κρυπτογραφήματα συμμετρικού κλειδιού μπορούν να συνθέσουν δυνατότερα κρυπτογραφήματα με απλούς μετασχηματισμούς.

5. Η κρυπτογραφία συμμετρικού κλειδιού έχει εκτεταμένη και πλούσια ιστορία.

### 2.6.2 Μειονεκτήματα Κρυπτογραφίας Συμμετρικού Κλειδιού

1. Κατά την επικοινωνία δύο οντοτήτων, το κλειδί πρέπει να παραμείνει μυστικό και στα δύο άκρα επικοινωνίας.

2. Σε δίκτυα μεγάλου μεγέθους, υπάρχουν πολλά ζευγάρια κλειδιών που πρέπει να διαχειριστούν.Αυτό συνεπάγεται ότι η αποτελεσματική διαχείριση κλειδιών απαιτεί τη χρήση έμπιστης τρίτης οντότητας (Trusted Third Party).

3. Κατά την επικοινωνία δύο οντοτήτων, το κλειδί πρέπει να αλλάζει πολύ συχνά, ενδεχομένως και πριν από κάθε εκ νέου επικοινωνία.

4. Οι μηχανισμοί ψηφιακής υπογραφής που προκύπτουν από την κρυπτογράφηση συμμετρικού κλειδιού, απαιτούν μεγαλύτερα κλειδιά για τη συνάρτηση δημόσιας επαλήθευσης και για τη χρήση έμπιστης τρίτης οντότητας.

### 2.6.3 Πλεονεκτήματα Κρυπτογραφίας Δημοσίου Κλειδιού

1. Μόνο το ιδιωτικό κλειδί πρέπει να διατηρείται μυστικό.Παρόλα αυτά, πρέπει να εγγυάται η αυθεντικότητα των δημοσίων κλειδιών.

2. Η διαχείριση των κλειδιών απαιτεί την παρουσία μιας έμπιστης τρίτης οντότητας, η οποία ανάλογα τον τρόπο λειτουργίας, μπορεί να χρειάζεται μόνο για off-line χρήση.

3. Ανάλογα με τον τρόπο λειτουργίας, ένα ζευγάρι ιδιωτικού/δημοσίου κλειδιού μπορεί να παραμείνει χωρίς αλλαγές για αξιοσημείωτο χρονικό διάστημα.

4. Πολλά σχήματα δημοσίου κλειδιού συνεπάγονται αποτελεσματικούς μηχανισμούς ψηφιακής υπογραφής.Το κλειδί που χρησιμοποιείται για την συνάρτηση δημόσιας επαλήθευσης είναι πολύ μικρότερο από το αντίστοιχο συμμετρικό κλειδί.

5. Σε ένα μεγάλο δίκτυο, ο αριθμός των απαραίτητων κλειδιών είναι σημαντικά μικρότερος από τον αντίστοιχο ενός σενάριου συμμετρικού κλειδιού.

### 2.6.4 Μειονεκτήματα Κρυπτογραφίας Δημοσίου Κλειδιού

1. Η ρυθμαπόδοση των πιο γνωστών σχημάτων κρυπτογραφίας δημοσίου κλειδιού είναι τάξεις μικρότερη από αυτήν των σχημάτων συμμετρικού κλειδιού.

2. Τα μεγέθη των κλειδιών είναι αρκετά μεγαλύτερα από τα αντίστοιχα εκείνων που απαιτούνται στην κρυπτογραφία συμμετρικού κλειδιού.

3. Δεν έχει βρεθεί σχήμα κρυπτογραφίας δημοσίου κλειδιού που να έχει αποδειχθεί ασφαλές(το ίδιο ισχύει και για τα block ciphers).Τα πιο αποτελεσματικά τέτοια σχήματα κρυπτογράφησης βασίζονται σε προβλήματα της θεωρίας αριθμών.

4. Η κρυπτογραφία δημοσίου κλειδιού δεν έχει τόσο εκτενή ιστορία όσο η συμμετρική κρυπτογραφία, αφού ανακαλύφθηκε τη δεκαετία του 1970.

### 2.6.5 Περίληψη της Σύγκρισης

Η συμμετρική κρυπτογραφία καθώς και η κρυπτογραφία δημοσίου κλειδιού έχουν συμπληρωματικά πλεονεκτήματα.Τα τρέχοντα κρυπτοσυστήματα προσπαθούν να εκμεταλευτούν τις δυνατότητες της καθεμίας και το σκεπτικό αυτό φαίνεται με το παρακάτω παράδειγμα.

Οι τεχνικές κρυπτογραφίας δημοσίου κλειδιού μπορούν να χρησιμοποιηθούν για τη δημιουργία ενός συμμετρικού κλειδιού που χρησιμοποιείται από τις οντότητες $A$ και $B$ που θέλουν να επικοινωνήσουν.Στο σενάριο αυτό οι οντότητες $A$ και $B$ εκμεταλεύονται τη μακρόχρονη διάρκεια ζωής των δημοσίων/ιδιωτικών κλειδιών του σχήματος κρυπτογραφίας δημοσίου κλειδιού και την αποδοτικότητα του σχήματος συμμετρικού κλειδιού.Από τη στιγμή που η κρυπτογράφηση των δεδομένων είναι συνήθως το πιο χρονοβόρο κομμάτι της συνολικής διαδικασίας, το σχήμα δημοσίου κλειδιού για την δημιουργία κλειδιού αποτελεί ένα μικρό κομμάτι της διαδικασίας κρυπτογράφησης μεταξύ των οντοτήτων $A$ και $B$.

Για να συνοψίσουμε, τα δύο βασικά συμπεράσματα της σύγκρισης αυτής είναι:

1. η κρυπτογραφία δημοσίου κλειδιού διευκολύνει τις διαδικασίες ψηφιακής υπογραφής και διαχείρισης κλειδιών

2. η κρυπτογραφία συμμετρικού κλειδιού είναι αποτελεσματική για κρυπτογράφηση δεδομένων και εφαρμογές που στοχεύουν στην ακεραιότητα των δεδομένων.

## 2.7 Γνωστά Εργαλεία Βασιζόμενα στην Κρυπτογράφηση Δημοσίου Κλειδιού

Παρακάτω παρουσιάζονται κάποιες εργαλεία που βασίζονται στην κρυπτογραφία δημοσίου κλειδιού η οποία συχνά αναφέρεται και ως ασύμμετρη κρυπτογράφηση.Όπως αναφέραμε και προηγουμένως, τα σχήματα κρυπτογράφησης δημοσίου κλειδιού είναι πολύ πιο αργά από τα αντίστοιχα της συμμετρικής κρυπτογράφησης και για το λόγο αυτό συνήθως χρησιμοποιούνται για τη μεταφορά κρυπτογραφικών κλειδιών.

### 2.7.1 RSA Κρυπτογράφηση

Το κρυπτοσύστημα RSA το οποίο ονομάστηκε έτσι από τα αρχικά των δημιουργών του (Rivest, Shamir και Adleman), είναι το πιο διαδεδομένο εργαλείο κρυπτογράφησης δημοσίου κλειδιού.Χρησιμοποιείται τόσο για μυστικότητα όσο

και για δημιουργία ψηφιακών υπογραφών.Η ασφάλεια του RSA βασίζεται στη δυσκολία του προβλήματος παραγοντοποίησης ακεραίων αριθμών.

### Περιγραφή του Πρωτοκόλλου

Με τους παρακάτω αλγόριθμους περιγράφονται οι διαδικασίες γέννησης κλειδιού για RSA κρυπτογράφηση και RSA κρυπτογράφηση.

---

**Αλγόριθμος 1** 'Γέννηση Κλειδιού για RSA κρυπτογράφηση'

---

Περίληψη: Ακολουθώντας τα παρακάτω βήματα κάθε οντότητα $A$ δημιουργεί ένα RSA δημόσιο κλειδί και το αντίστοιχο ιδιωτικό.

1. Γέννηση δύο μεγάλων, διακριτών, τυχαίων, πρώτων ακεραίων αριθμών $p$ και $q$.
2. Υπολογισμός $n = pq$ και $\varphi = (p-1)(q-1)$.
3. Επιλογή ενός τυχαίου ακεραίου $e$, $1 < e < \varphi$, έτσι ώστε $gcd(e, \varphi) = 1$.
4. Χρήση του αλγορίθμου Extended Euclidean Algorithm για τον υπολογισμό ενός μοναδικού ακεραίου $d$, $1 < d < \varphi$, έτσι ώστε $ed = 1(mod\varphi)$.
5. Το δημόσιο κλειδί του $A$ είναι το $(n, e)$ και το ιδιωτικό το $d$.

---

Οι ακέραιοι αριθμοί $e$ και $d$ στο σχήμα γέννησης κλειδίου RSA ονομάζονται encryption exponent και decryption exponent αντίστοιχα ενώ το $n$ ονομάζεται modulus.

**Αλγόριθμος 2** 'RSA κρυπτογράφηση δημοσίου κλειδιού'

Περίληψη: Η οντότητα $B$ κρυπτογραφεί ένα μήνυμα $m$, το οποίο αποκρυπτογραφεί η οντότητα $A$.

1. Κρυπτογράφηση:Η οντότητα $B$ εκτελεί τα παρακάτω βήματα:

   Αποκτά το αυθεντικό δημόσιο κλειδί $(n, e)$ της οντότητας $A$.

   Αναπαριστά το μήνυμα ως έναν ακέραιο στο διάστημα $[o, n-1]$.

   Υπολογίζει το $c = m^e(modn)$

   Στέλνει στον $A$ το κρυπτογράφημα $c$.

2. Αποκρυπτογράφηση: Η οντότητα $A$ εκτελεί το ακόλουθο βήμα για να υπολογίσει το αρχικό μήνυμα $m$ από το κρυπτογράφημα $c$.

   Χρησιμοποιεί το ιδιωτικό κλειδί $d$ και υπολογίζει το μήνυμα $m = c^d(modn)$.

---

**Παράδειγμα**:

**Γέννηση Κλειδιού**:Η οντότητα $A$ επιλέγει τους πρώτους αριθμούς $p = 2357$, $q = 2551$ και υπολογίζει το $n = pq = 6012707$ και το $\varphi = (p-1)(q-1) = 6007800$.Στη συνέχεια, επιλέγει τον αριθμό $e = 3674911$ και χρησιμοποιώντας τον ευκλείδιο αλγόριθμο βρίσκει το $d = 422191$ έτσι ώστε $ed = 1(mod\varphi)$.Το δημόσιο κλειδί του $A$ είναι το ζευγάρι $(n, e) = (6012707, 3674911)$ και το ιδιωτικό του είναι το $d = 422191$.

**Κρυπτογράφηση**:Για την κρυπτογράφηση του μηνύματος $m = 5234673$ η οντότητα $B$ υπολογίζει:

$$c = m^e(modn) = 5234673^{3674911}(mod6012707) = 3650502.$$

**Αποκρυπτογράφηση**:Για την αποκρυπτογράφηση του $c$ η οντότητα $A$ υπολογίζει:

$$c^d(modn) = 3650502^{422191}(mod6012707) = 5234673.$$

### Ασφάλεια του RSA

Η ασφάλεια του κρυπτοσυστήματος RSA βασίζεται σε δύο μαθηματικά προβλήματα: αυτό της παραγοντοποίησης μεγάλων ακεραίων και το RSA πρόβλημα.Η πλήρης αποκρυπτογράφηση ενός RSA κρυπτογραφήματος θεωρείται αδύνατη αν υποθέσει κανείς ότι τα δύο προβλήματα είναι *δύσκολα* (δεν υπάρχει αλγόριθμος που να τα λύνει).

Το πρόβλημα RSA ορίζεται ως η απόπειρα εύρεσης των $e$-οστών ριζών modulo ενός σύνθετου αριθμού $n$, έτσι ώστε να υπολογιστεί μια τιμή $m$ για την οποία $c = m^e(modn)$, όπου $(n, e)$ είναι ένα RSA δημόσιο κλειδί και $c$ ένα κρυπτογράφημα.Η πιο ελπιδοφόρος προσέγγιση επίλυσης του RSA προβλήματος είναι αυτή της παραγοντοποίησης του modulus $n$.Με την ικανότητα να ανακτά πρώτους παράγοντες, δηλαδή δεδομένου του $n$ να βρίσκει τα $p, q$, ένας επιτιθέμενος μπορεί υπολογίζοντας το $(p-1)(q-1)$ να βρει το μυστικό κλειδί $d$ από το δημόσιο κλειδί $(n, e)$ και στη συνέχεια να αποκρυπτογραφεί μηνύματα εκτελώντας την διαδικασία αποκρυπτογράφησης.Μέχρι στιγμής δεν έχει βρεθεί μέθοδος που να παραγοντοποιεί ακέραιους αριθμούς σε πολυωνυμικό χρόνο.Βέβαια, ούτε έχει αποδειχθεί ότι δεν υπάρχει τέτοια μέθοδος.

Τυπικά, τα κλειδιά RSA έχουν μέγεθος 1024-2048 bits.Οι ειδικοί εκτιμούν ότι τα κλειδιά μεγέθους 1024 bits θα είναι εύθραυστα στο κοντινό μέλλον και προτείνουν τη χρήση κλειδιών μεγέθους 4096 bits.Γενικότερα, το RSA θεωρείται ασφαλές εάν το $n$ είναι αρκετά μεγάλο.Παραδείγματος χάριν, αναφέρουμε ότι αν το $n$ έχει μέγεθος 300 bits μπορεί να παραγοντοποιηθεί σε μερικές ώρες σε έναν προσωπικό υπολογιστή με λογισμικό που είναι ήδη διαθέσιμο.Τα κλειδιά RSA μήκους 512 bits πλέον μπορούν να παραγοντοποιηθούν σε μερικές εβδομάδες και μια θεωρητική μηχανή υλικού που ονομάζεται TWIRL θέτει υπο αμφισβήτηση τα κλειδιά 1024 bits.Σήμερα, το $n$ συνιστάται να έχει μέγεθος 2048 bits.

### 2.7.2 Το σχήμα El Gamal

Το κρυπτοσύστημα El Gamal βασίζεται στο πρωτόκολλο συμφωνίας κλειδιών Diffie Hellman.Η ασφάλεια του βασίζεται στο πρόβλημα του διακριτού λογαρίθμου και στο πρόβλημα Diffie Hellman.Το πρόβλημα διακριτού λογαρίθμου διατυπώνεται ως εξής: Δεδομένου ενός πρώτου αριθμού $p$, ενός γεννήτορα $a$ του $Z_p^*$ και ενός στοιχείου $b \in Z_p^*$, να βρεθεί ένας ακέραιος $x$, με $0 \leq x \leq p-2$, έτσι ώστε $a^x = b(modp)$.

#### Περιγραφή του Πρωτοκόλλου

Με τους παρακάτω αλγόριθμους περιγράφονται οι διαδικασίες γέννησης κλειδιού για El Gamal κρυπτογράφηση και El Gamal κρυπτογράφηση δημοσίου κλειδιού.

**Αλγόριθμος 3** 'Γέννηση Κλειδιού για El Gamal κρυπτογράφηση'

Περίληψη: Ακολουθώντας τα παρακάτω βήματα κάθε οντότητα $A$ δημιουργεί ένα El Gamal δημόσιο κλειδί και το αντίστοιχο ιδιωτικό.

1. Γέννηση ενός μεγάλου τυχαίου πρώτου αριθμού και ενός γεννήτορα $g$ της πολλαπλασιαστικής ομάδας $Z_p^*$ των ακεραίων αριθμών modulo $p$.

2. Επιλογή ενός τυχαίου ακεραίου αριθμού $a$, με $1 \leq a \leq p-2$, και υπολογισμός του $g^a(modp)$.

3. Το δημόσιο κλειδί του $A$ είναι το $(p, g, g^a)$ και το ιδιωτικό είναι το $a$.

---

**Αλγόριθμος 4** 'El Gamal κρυπτογράφηση δημοσίου κλειδιού'

Περίληψη: Η οντότητα $B$ κρυπτογραφεί ένα μήνυμα $m$, το οποίο αποκρυπτογραφεί η οντότητα $A$.

1. Κρυπτογράφηση:Η οντότητα $B$ εκτελεί τα παρακάτω βήματα:

   Αποκτά το δημόσιο κλειδί $(p, g, g^a)$ της οντότητας $A$.

   Αναπαριστά το μήνυμα ως έναν ακέραιο $m$ στο διάστημα $[0, ..., p-1]$.

   Διαλέγει έναν τυχαίο ακέραιο αριθμό $k$, όπου $1 \leq k \leq p-2$.

   Υπολογίζει $\gamma = g^k(modp)$ και $\delta = m(g^a)^k(modp)$.

   Στέλνει στον $A$ το κρυπτογράφημα $c = (\gamma, \delta)$.

2. Αποκρυπτογράφηση: Η οντότητα $A$ εκτελεί το ακόλουθο βήμα για να υπολογίσει το αρχικό μήνυμα $m$ από το κρυπτογράφημα $c$.

   Χρησιμοποιεί το ιδιωτικό κλειδί $a$ και υπολογίζει $\gamma^{p-1-a}$ ($\gamma^{p-1-a} = \gamma^{-a} = g^{-ak}$).

   Ανακτά το $m$ υπολογίζοντας $(\gamma^{-a})\delta(modp)$.

---

**Παράδειγμα**:

**Γέννηση Κλειδιού**:Η οντότητα $A$ επιλέγει τον πρώτο αριθμό $p = 2357$ και τον γεννήτορα $g = 2$ του $Z_{2357}^*$.Διαλέγει το ιδιωτικό κλειδί $a = 1751$ και υπολογίζει:

$$g^a(modp) = 2^{1751}(mod2357) = 1185$$

Το δημόσιο κλειδί του $A$ είναι το $(p = 2357, g = 2, g^a = 1185)$.

**Κρυπτογράφηση**:Για την κρυπτογράφηση του μηνύματος $m = 2035$, ο $B$ υπολογίζει έναν τυχαίο ακέραιο $k = 1520$ και υπολογίζει:

$$\gamma = 2^{1520}(mod2357) = 1430 \text{ και } \delta = 2035 * 1185^{1520}(mod2357) = 697.$$

Ο $B$ στέλνει στον $A$, $\gamma = 1430$ και $\delta = 697$.

**Αποκρυπτογράφηση**:Για την αποκρυπτογράφηση ο $A$ υπολογίζει:

$$\gamma^{p-1-a} = 1430^{605}(mod2357) = 872$$

και ανακτά το $m$ υπολογίζοντας:

$$m = 872 \times 697(mod2357) = 2035.$$

### Ασφάλεια του El Gamal

Η ασφάλεια του σχήματος κρυπτογράφησης δημοσίου κλειδιού El Gamal, βασίζεται στη δυσκολία επίλυσης του προβλήματος διακριτού λογαρίθμου στο πεδίο $Z_p^*$.Η προσπάθεια ανάκτησης του $m$ δεδομένων των $p, a, a^a, \gamma, \delta$ είναι υπολογιστικά ισοδύναμα δύσκολη με την επίλυση του προβλήματος Diffie-Hellman (αναλύεται παρακάτω).Το σχήμα κρυπτογράφησης El Gamal μπορεί να θεωρηθεί ως μια διαδικασίας ανταλλαγής κλειδιών Diffie-Hellman για τον καθορισμό του κλειδιού $a^{ak}$ και στην συνέχεια της κρυπτογράφησης του μηνύματος πολλαπλασιάζοντάς το με αυτό το κλειδί.Είναι πολύ σημαντικό να χρησιμοποιούνται διαφορετικοί τυχαίοι ακέραιοι $k$ για την κρυπτογράφηση διαφορετικών μηνυμάτων.Για παράδειγμα, ας θεωρήσουμε ότι ο ίδιος αριθμός $k$ χρησιμοποιείται για την κρυπτογράφηση δύο μηνυμάτων $m1, m2$.Στην περίπτωση αυτή τα κρυπτογραφημένα ζευγάρια είναι τα $(\gamma_1, \delta_1)$ και $(\gamma_2, \delta_2)$.Έτσι, $\delta_1/\delta_2 = m_1/m_2$ και το $m_2$ θα μπορούσε να υπολογιστεί εύκολα εαν το $m_1$ είναι γνωστό.

Δεδομένης της προόδου που έχει γίνει τα τελευταία χρόνια για την επίλυση του προβλήματος διακριτού λογαρίθμου στο πεδίο $Z_p^*$, κατάλληλα κλειδιά για El Gamal κρυπτογράφηση θεωρούνται αυτά που έχουν μέγεθος τουλάχιστον 1024 bits.

## 2.7.3 Πρωτόκολλο Συμφωνίας Κλειδιών Diffie-Hellman

Το πρωτόκολλο Diffie-Hellman αποτελεί την πρώτη λύση στο πρόβλημα της διανομής κλειδιών.Επιτρέπει σε δύο οντότητες που δεν έχουν επικοινωνήσει προηγουμένως ή που δε μοιράζονται κρυπτογραφικό υλικό, να δημιουργήσουν ένα κοινό μυστικό ανταλλάσοντας μηνύματα πάνω από ένα ανασφαλές κανάλι.Η ασφάλεια του σχήματος αυτού βασίζεται στο πρόβλημα Diffie-Hellman και στο σχετικό πρόβλημα του διακριτού λογαρίθμου.Το πρόβλημα Diffie-Hellman ορίζεται ως εξής: δεδομένου ενός πρώτου αριθμού $p$, ενός γεννήτορα $a$ του πεδίου $Z_p^*$ και των στοιχείων $a^a(modp)$ και $a^b(modp)$, να βρεθεί το $a^{ab}(modp)$.Το σχήμα Diffie-Hellman συνήθως χρησιμοποιείται για τη δημιουργία ενός μυστικού κλειδιού μεταξύ δύο οντοτήτων, το οποίο χρησιμοποιούν στη συνέχεια για

κρυπτογράφηση με κάποιο αλγόριθμο συμμετρικού κλειδιού.Η βασική μορφή του σχήματος αυτού που παρουσιάζεται παρακάτω, προστατεύει το κοινό κλειδί που δημιουργείται από παθητικούς επιτιθέμενους (ωτακουστές) αλλά όχι από ενεργούς.Επιπλέον, το σχήμα αυτό δεν παρέχει αυθεντικότητα ως προς τις οντότητες και ως προς τα κλειδιά.

### Περιγραφή του Πρωτοκόλλου

Παρακάτω παρουσιάζεται η βασική έκδοση του πρωτοκόλλου συμφωνίας κλειδιών Diffie-Hellman.

---

**Αλγόριθμος 5** 'Πρωτόκολλο συμφωνίας κλειδιών Diffie-Hellman'

---

**Περίληψη**: Οι οντότητες $A$,$B$ ανταλλάσουν από ένα μήνυμα πάνω από ένα ανοιχτό κανάλι.
**Αποτέλεσμα**: Ένα κοινό μυστικό $K$ το οποίο γνωρίζουν και οι δυο οντότητες $A$,$B$.

1. Φάση Συμφωνίας Παραμέτρων: Διαλέγονται και δημοσιεύονται ένας κατάλληλος πρώτος αριθμός $p$ και ένας γεννήτορας $a$ του $Z_p^*$ με $2 \leq a \leq p-2$.

2. Μηνύματα Πρωτοκόλλου:

$$A \rightarrow B : a^x (mod p) \ (1)$$
$$B \rightarrow A : a^y (mod p) \ (2)$$

3. Πράξεις Πρωτοκόλλου:Οι οντότητες $A, B$ εκτελούν τα παρακάτω βήματα κάθε φορά που απαιτείται ένα διαμοιραζόμενο κλειδί.

   Ο $A$ διαλέγει ένα τυχαίο μυστικό $x$, όπου $1 \leq x \leq p-2$, και στέλνει στον $B$ το μήνυμα (1).

   Ο $B$ διαλέγει ένα τυχαίο μυστικό $y$, όπου $1 \leq y \leq p-2$, και στέλνει στον $A$ το μήνυμα (2).

   Ο $B$ λαμβάνει το $a^x$ και υπολογίζει το διαμοιραζόμενο μυστικό $K = (a^x)^y (mod p)$.

   Ο $A$ λαμβάνει το $a^y$ και υπολογίζει το διαμοιραζόμενο μυστικό $K = (a^y)^x (mod p)$.

---

**Παράδειγμα**:

1. Οι οντότητες $A, B$ συμφωνούν στην χρήση του πρώτου αριθμού $p = 23$ και του γεννήτορα $a = 5$.

2. Η οντότητα $A$ διαλέγει ένα μυστικό ακέραιο $x = 6$ και στέλνει στην οντότητα $B$ το μήνυμα $M_A$:

   $M_A = 5^6(mod 23) = 8.$

3. Η οντότητα $B$ διαλέγει ένα μυστικό ακέραιο $y = 15$ και στέλνει στην οντότητα $A$ το μήνυμα $M_B$:

   $M_B = 5^{15}(mod 23) = 19.$

4. Η οντότητα $A$ υπολογίζει το κοινό μυστικό $K = (M_B)^x(mod p)$.

   $19^6(mod 23) = 2.$

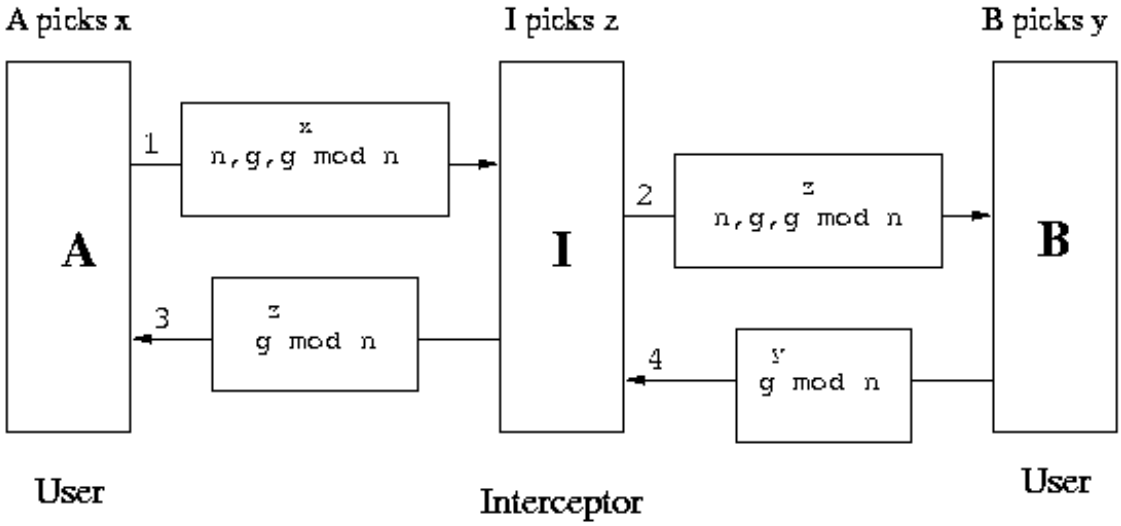
5. Η οντότητα $B$ υπολογίζει το κοινό μυστικό $K = (M_A)^y(mod p)$.

   $8^{15}(mod 23) = 2.$

### Ασφάλεια του Diffie-Hellman

Το πρωτόκολλο Diffie-Hellman είναι ασφαλές απέναντι σε ωτακουστές αρκεί οι παράμετροι $p$ και $a$ να επιλεχθούν κατάλληλα (ο αριθμός $p$ πρέπει να είναι αρκετά μεγάλος).Για να μπορέσει ένας ωτακουστής να υποκλέψει το κοινό μυστικό θα πρέπει να λύσει το πρόβλημα Diffie-Hellman και κατά επέκταση το πρόβλημα διακριτού λογαρίθμου.Αν οι οντότητες που επικοινωνούν χρησιμοποιούν γεννήτριες τυχαίων αριθμών των οποίων τα αποτελέσματα δεν είναι εντελώς τυχαία και μπορούν να προβλεφθούν σε κάποιο βαθμό, τα πράματα διευκολύνονται για τον ωτακουστή.Όπως παρατηρούμε τα μυστικά $x, y$ στο τέλος της επικοινωνίας παραμερίζονται.Για αυτό το λόγο, λέμε ότι το πρωτόκολλο Diffie-Hellman επιτυγχάνει τέλεια προς τα εμπρός μυστικότητα, αφού μακροπρόθεσμα δεν υπάρχει κάποιο μυστικό που μπορεί να αποκαλυφθεί.

Ο αλγόριθμος Diffie-Hellman έχει μια βασική αδυναμία.Δεν παρέχει πιστοποίηση μεταξύ των οντοτήτων που επικοινωνούν και έτσι είναι ευάλωτο στην επίθεση τύπου man in the middle.Στην επίθεση αυτή, μια οντότητα $C$ μπορεί να εγκαθιδρύσει δύο επικοινωνίες τύπου Diffie-Hellman, μία με την οντότητα $A$ και μια με την οντότητα $B$.Στην επικοινωνία με την οντότητα $A$ παρουσιάζεται ως οντότητα $B$ και στην επικοινωνία με την οντότητα $B$ παρουσιάζεται ως οντότητα $A$.Με τον τρόπο αυτό, λαμβάνει τα δεδομένα που ανταλλάσονται μεταξύ των οντοτήτων $A, B$, τα αποκρυπτογραφεί, τα διαβάζει και τα επανακρυπτογραφεί.Καταλαβαίνουμε λοιπόν, ότι απαιτείται μια μέθοδος πιστοποίησης των οντοτήτων που επικοινωνούν για την αντιμετώπιση αυτής της επίθεσης.




Σχήμα 2.7: Η επίθεση τύπου man in the middle.

## 2.8 Ψηφιακές Υπογραφές

Μια κρυπτογραφική ιδιότητα που παίζει θεμελιώδη ρόλο στην αυθεντικότητα, στην εξουσιοδότηση και στη μη-αποποίηση είναι η ψηφιακή υπογραφή.Ο σκοπός χρήσης των ψηφιακών υπογραφών είναι η ύπαρξη ενός μέσου με το οποίο μια οντότητα να δεσμεύεται με ένα κομμάτι πληροφορίας.Η διαδικασία υπογραφής περιλαμβάνει τη μετατροπή ενός μηνύματος και κάποιων μυστικών πληροφοριών που κατέχει η οντότητα σε μια ετικέτα που ονομάζεται υπογραφή.Η περιγραφή ενός γενικού σχήματος ψηφιακής υπογραφής παρουσιάζεται παρακάτω.

### 2.8.1 Ορολογία

- $M$ είναι το σύνολο των μηνυμάτων που μπορούν να υπογραφούν.
- $S$ είναι ένα σύνολο στοιχείων που ονομάζονται υπογραφές και πιθανότατα είναι δυαδικά αλφαριθμητικά σταθερού μήκους.

- $S_A$ είναι ένας μετασχηματισμός από το σύνολο μηνυμάτων $M$ στο σύνολο υπογραφών $S$ και ονομάζεται μετασχηματισμός υπογραφής για την οντότητα $A$.Ο μετασχηματισμός $S_A$ διατηρείται μυστικός από την $A$ και χρησιμοποιείται για τη δημιουργία υπογραφών των μηνυμάτων του $M$.

- $V_A$ είναι ένας μετασχηματισμός από το σύνολο $M \times S$ στο σύνολο $\{true, false\}$.Ο $V_A$ ονομάζεται μετασχηματισμός επαλήθευσης τν υπογραφών της οντότητας $A$ και είναι δημοσίως γνωστός έτσι ώστε άλλες οντότητες να μπορούν να επαληθεύσουν τις υπογραφές που δημιουργεί η οντότητα αυτή.

Οι μετασχηματισμοί $S_A$ και $V_A$ αποτελούν ένα σχήμα ψηφιακής υπογραφής για την οντότητα $A$.Συχνά χρησιμοποιείται ο όρος *μηχανισμός ψηφιακής υπογραφής*.

### 2.8.2 Διαδικασία Υπογραφής

Η οντότητα $A$ (ο υπογραφών) δημιουργεί μια υπογραφή για το μήνυμα $m \in M$ κάνοντας τα εξής:

1. Υπολογίζει το $s = S_A(m)$.

2. Μεταδίδει το ζευγάρι $(m, s)$.Το $s$ είναι η υπογραφή του μηνύματος $m$.

### 2.8.3 Διαδικασία Επαλήθευσης

Για να επαληθεύσει μια οντότητα $B$ (επιβεβαιωτής) ότι η υπογραφή $s$ στο μήνυμα $m$ δημιουργήθηκε από την οντότητα $A$ εκτελεί τα ακόλουθα βήματα:

1. Αποκτά τη συνάρτηση επαλήθευσης $V_A$ της οντότητας $A$.

2. Υπολογίζει το $u = V_A(m, s)$.

3. Δέχεται το μήνυμα εάν $u = true$, δηλαδή η υπογραφή δημιουργήθηκε από την οντότητα $A$ και το απορίπτει εάν $u = false$.

Οι μετασχηματισμοί $S_A$ και $V_A$ τυπικά χαρακτηρίζονται από ένα κλειδί.Υπάρχει μία κλάση αλγορίθμων υπογραφής και επαλήθευσης, οι οποίοι είναι δημοσίως γνωστοί.Κάθε αλγόριθμος μέσα στην κλάση αυτή αναγνωρίζεται από ένα κλειδί.Έτσι ο αλγόριθμος υπογραφής $S_A$ της οντότητας $A$ καθορίζεται από ένα κλειδί $k_A$ το οποίο παραμένει μυστικό.Αντίστοιχα, ο αλγόριθμος επαλήθευσης $V_A$ της οντότητας $A$ καθορίζεται από ένα κλειδί $l_A$ το οποίο είναι κοινώς γνωστό.

### 2.8.4 Απαραίτητες Ιδιότητες των Συναρτήσεων Υπογραφής και Επαλήθευσης

Οι συναρτήσεις υπογραφής και επαλήθευσης πρέπει να ικανοποιούν δύο βασικές ιδιότητες:

1. Το $s$ είναι μια έγκυρη υπογραφή της οντότητας $A$ στο μήνυμα $m$ αν και μόνο αν $V_A(m, s) = true$.

2. Είναι υπολογιστικά αδύνατο για οποιαδήποτε οντότητα πλην της $A$, να βρει ένα $s \in S$, για οποιοδήποτε $m \in M$, έτσι ώστε $V_A(m, s) = true$.

Μέχρι στιγμής κανένας δεν έχει αποδείξει επίσημα ότι τα υπάρχοντα σχήματα ψηφιακών υπογραφών ικανοποιούν τη δεύτερη ιδιότητα.Μια μεγάλη ποικιλία μηχανισμών ψηφιακής υπογραφής προκύπτει από τα σχήματα κρυπτογράφησης δημοσίου κλειδιού.

### 2.8.5 Ψηφιακές Υπογραφές από Αναστρέψιμη Κρυπτογραφία Δημοσίου Κλειδιού

Ας υποθέσουμε ότι $E_e$ είναι ένας μετασχηματισμός κρυπτογράφησης με χώρο μηνύματος $M$ και χώρο κρυπτογραφημάτων $C$, όπου $M = C$.Εάν $D_d$ είναι ο αντίστοιχος μετασχηματισμός αποκρυπτογράφησης τότε αφού τα $E_e$ και $D_d$ είναι μεταθέσεις ισχύει:

$$D_d(E_e(m)) = E_e(D_d(m)) = m, \text{ για κάθε } m \in M.$$

Ένα τέτοιο σχήμα κρυπτογράφησης ονομάζεται αναστρέψιμο.Τονίζουμε ότι θα πρέπει να ισχύει $M = C$ ώστε η παραπάνω ισότητα να ισχύει για κάθε $m \in M$.

#### Κατασκευή Σχήματος Ψηφιακής Υπογραφής

1. $M$ είναι ο χώρος μηνυμάτων για το σχήμα υπογραφής.

2. $C = M$ είναι ο χώρος υπογραφών $S$.

3. $(e, d)$ είναι ένα ζευγάρι κλειδιών για το σχήμα κρυπτογράφησης δημοσίου κλειδιού.

4. Ως συνάρτηση υπογραφής ορίζεται η $D_d$ δηλαδή $S_A = D_d$.Έτσι, η υπογραφή για ένα μήνυμα $m \in M$ είναι η $s = D_d(m)$.

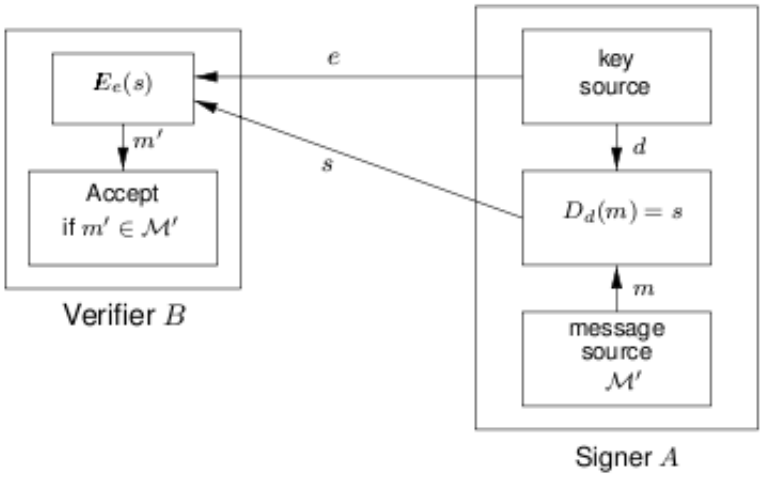
5. Η συνάρτηση επαλήθευσης ορίζεται ως εξής:

$$V_A(m,s) = \begin{cases} true, & if E_e(s) = m \\ false, & otherwise \end{cases}$$

Το παραπάνω σχήμα μπορεί να απλοποιηθεί περαιτέρω, εάν ο $A$ υπογράφει μόνο μηνύματα που έχουν συγκεκριμένη δομή (η οποία είναι κοινώς γνωστή).Ας θεωρήσουμε το $M'$ ως ένα υποσύνολο του $M$, όπου τα στοιχεία του $M'$ έχουν κάποια ειδική δομή.Αφού ο $A$ υπογράφει μόνο μηνύματα του συνόλου $M'$, αυτά αναγνωρίζονται εύκολα από έναν επαληθευτή.Η συνάρτηση επαλήθευσης ορίζεται ως εξής:

$$V_A(m,s) = \begin{cases} true, & if E_e(s) \in M' \\ false, & otherwise \end{cases}$$

Στο σενάριο αυτό ο $A$ χρειάζεται να μεταδόσει μόνο την υπογραφή $s$ καθώς το μήνυμα $m$ μπορεί να ανακτηθεί εφαρμόζοντας τη συνάρτηση επαλήθευσης.Ένα τέτοιο σχήμα ονομάζεται *σχήμα ψηφιακής υπογραφής με ανάκτηση μηνύματος*.Η παρακάτω εικόνα παρουσιάζει πως χρησιμοποιείται η συνάρτηση υπογραφής.Η επιλογή μηνυμάτων ειδικής δομής αναφέρεται ως επιλογή μηνυμάτων με πλεονασμό.




Σχήμα 2.8: Σχήμα ψηφιακής υπογραφής με ανάκτηση μηνύματος.

Οι απλοποιήσεις που κάναμε στο παραπάνω σχήμα έχουν ένα βασικό μειονέκτημα.Ας υποθέσουμε ότι η οντότητα $B$ επιλέγει ένα τυχαίο στοιχείο $s \in S$ σαν υπογραφή και εφαρμόζει το μετασχηματισμό $E_e$ για να υπολογίσει το $u = E_e(s)$, καθώς $S = M$ και ο $E_e$ είναι κοινώς γνωστός.Ο $B$ μπορεί να μεταδόσει το ζευγάρι $(m, s)$ όπου το μήνυμα $m = u$.Έτσι, η υπογραφή $s$ θα

επαληθευτεί ότι δημιουργήθηκε από την οντότητα $A$ για το μήνυμα $m$, κάτι που όμως δε συνέβη στην πραγματικότητα.Με τον παραπάνω τρόπο ο $B$ έχει πλαστογραφήσει την υπογραφή του $A$ γεγονός που έρχεται σε αντίθεση με τις ιδιότητες των συναρτήσεων υπογραφής και επαλήθευσης.

Παρά το γέγονος ότι τα σχήματα ψηφιακής υπογραφής που βασίζονται στην αντιστρέψιμη κρυπτογραφία δημοσίου κλειδιού είναι ελκυστικά, απαιτούν σαν βασικό στοιχείο ένα μηχανισμό κρυπτογράφησης.Σε περιπτώσεις που η εφαρμογή ενός μηχανισμού κρυπτογράφησης είναι χρονοβόρα διαδικασία, τέτοια σχήματα ψηφιακής εφαρμογής δεν είναι κατάλληλα.

# Κεφάλαιο 3

# Οι Ελλειπτικές Καμπύλες στην Κρυπτογραφία

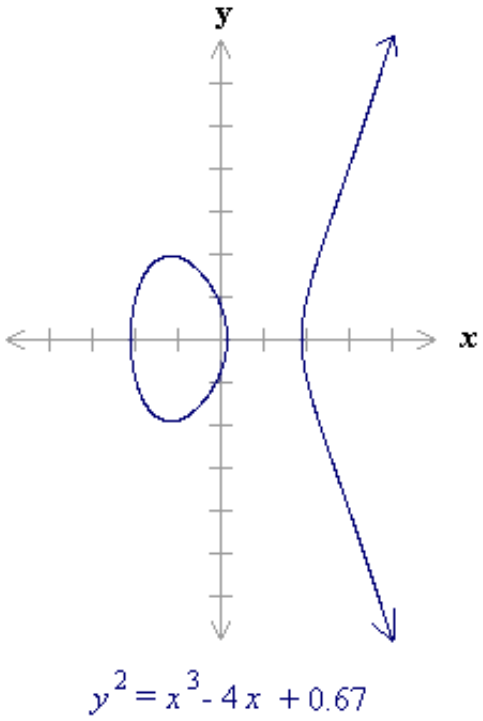
## 3.1 Θεωρία Ελλειπτικών Καμπυλών

Οι ελλειπτικές καμπύλες ως αλγεβρικές/γεωμετρικές οντότητες έχουν μελετηθεί εκτενώς τα τελευταία 150 χρόνια και από τις μελέτες αυτές έχει προκύψει πλούσια θεωρία.

Πολλά κρυπτοσυστήματα συχνά απαιτούν τη χρήση αλγεβρικών ομάδων και οι ελλειπτικές καμπύλες μπορούν να χρησιμοποιηθούν για τη δημιουργία τέτοιων ομάδων.Μία ομάδα είναι ένα σύνολο στοιχείων με ορισμένες αριθμητικές πράξεις πάνω σε αυτά.Για τις ομάδες ελλειπτικών καμπυλών οι πράξεις αυτές ορίζονται γεωμετρικά.Εισάγοντας πιο αυστηρές ιδιότητες στα στοιχεία μιας ομάδας,δημιουργείται ένα υποκείμενο πεδίο για την ομάδα της ελλειπτικής καμπύλης.Οι ελλειπτικές καμπύλες εξετάζονται πρώτα πάνω από το πεδίο των πραγματικών αριθμών έτσι ώστε να απεικονιστούν οι γεωμετρικές ιδιότητες των ελλειπτικών ομάδων.Εν συνεχεία εξετάζονται πάνω από τα δυαδικά πεδία $F_{2^m}$ καθώς και πάνω από τα πρώτα πεδία $F_p$.

### 3.1.1 Ομάδες Ελλειπτικών Καμπυλών Ορισμένες Πάνω από το Πεδίο των Πραγματικών Αριθμών

Μία ελλειπτική καμπύλη πάνω από το πεδίο των πραγματικών αριθμών ορίζεται ως ένα σύνολο σημείων $(x, y)$ που ικανοποιούν την εξίσωση της μορφής $y^2 = x^3 + ax + b$ όπου τα $x, y$ και $a, b$ είναι πραγματικοί αριθμοί.Κάθε επιλογή των αριθμών $a, b$ δημιουργεί και μία διαφορετική ελλειπτική καμπύλη.Για παράδειγμα εάν $a = -4$ και $b = 0.67$ τότε δημιουργείται η καμπύλη με εξίσωση $y^2 = x^3 - 4x + 0.67$ της οποίας η γραφική απεικόνιση φαίνεται παρακάτω.




Σχήμα 3.1: Η ελλειπτική καμπύλη με εξίσωση $y^2 = x^3 - 4x + 0.67$.

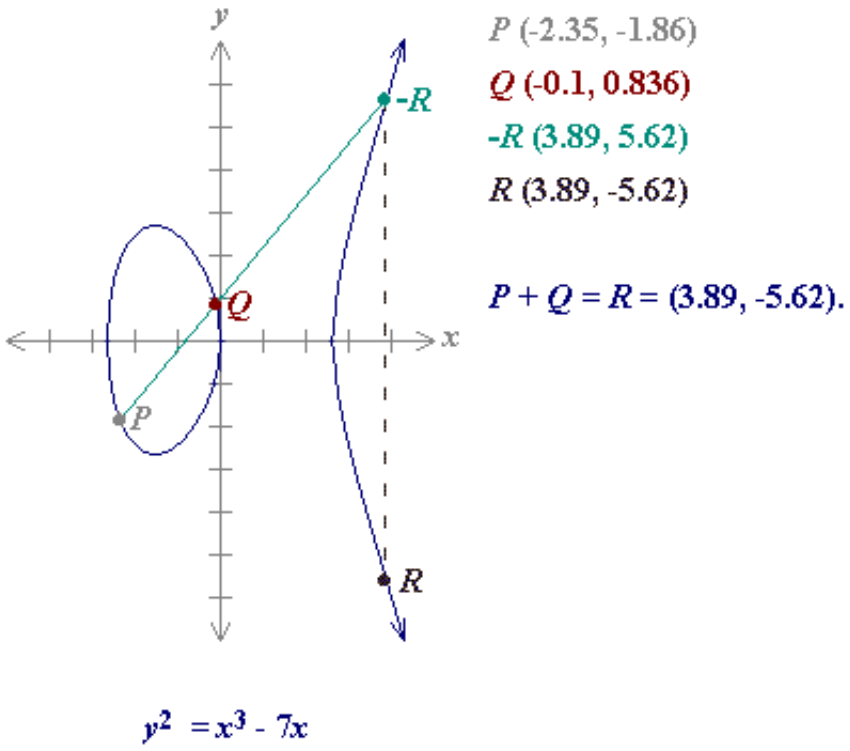
Εφόσον το $x^3 + ax + b$ δεν περιέχει επαναλαμβανόμενους παράγοντες ή ισοδύναμα ισχύει ότι $4a^3 + 27b^2 \neq 0$ τότε η εξίσωση $y^2 = x^3 + ax + b$ μπορεί να χρησιμοποιηθεί για να δημιουργήσει μία ομάδα.Μία ομάδα ελλειπτικής καμπύλης αποτελείται από τα σημεία της αντίστοιχης ελλειπτικής καμπύλης μαζί με ένα ειδικό σημείο $O$ που ονομάζεται σημείο στο *άπειρο*.Το πλήθος των σημείων μιας ελλειπτικής καμπύλης ονομάζεται τάξη της ελλειπτικής καμπύλης.

### Πρόσθεση Σημείων μιας Ελλειπτικής Καμπύλης (Γεωμετρική Προσέγγιση)

Οι ομάδες ελλειπτικών καμπυλών είναι προσθετικές ομάδες δηλαδή η βασική τους λειτουργία είναι η πρόσθεση.Η πρόσθεση δύο σημείων μιας ελλειπτικής καμπύλης ορίζεται γεωμετρικά.Για κάθε σημείο $P(x, y)$ μιας ελλειπτικής καμπύλης το σημείο $-P$ ορίζεται ως η συμμετρία του $P$ ως προς τον άξονα $x$ δηλαδή είναι το σημείο $(x, -y)$. Αναφέρουμε ότι για κάθε σημείο $P$ μιας ελλειπτικής καμπύλης το σημείο $-P$ βρίσκεται επίσης πάνω στην καμπύλη.

Για την πρόσθεση δύο δεδομένων σημείων $P, Q$ μιας ελλειπτικής καμπύλης

σχεδιάζεται μια ευθεία γραμμή μεταξύ των σημείων.Η γραμμή αυτή τέμνει την καμπύλη σε ένα τρίτο σημείο του οποίου το συμμετρικό ως προς τον άξονα $x$ είναι το αποτέλεσμα της πρόσθεσης.Όπως φαίνεται και παρακάτω $P + Q = R$.

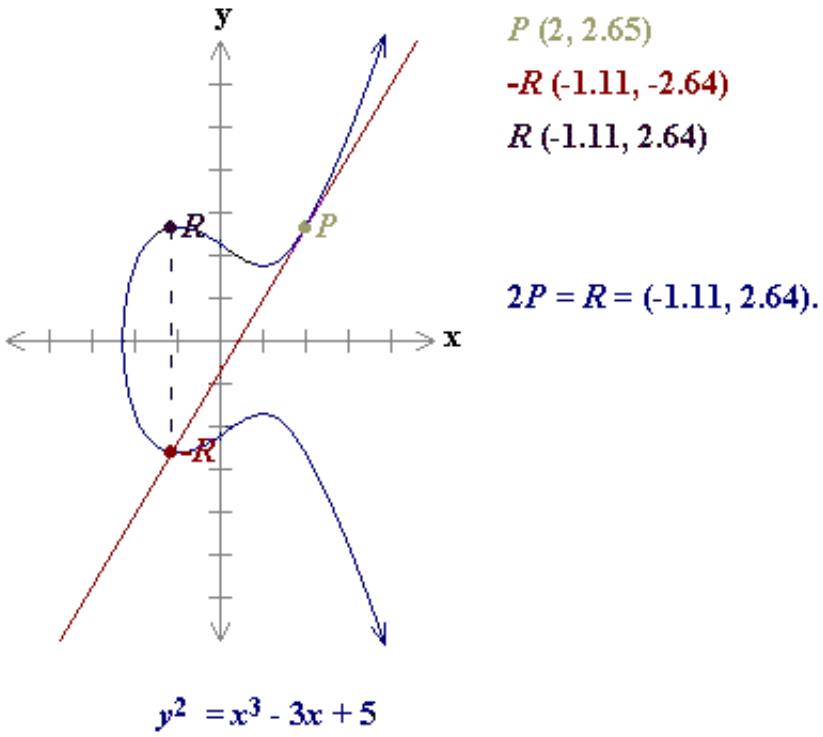



Σχήμα 3.2: Πρόσθεση δύο σημείων $P, Q$ μιας ελλειπτικής καμπύλης.

Η ευθεία γραμμή που σχεδιάζεται ανάμεσα στο σημείο $P$ και στο αντίθετό του $-P$ δεν τέμνει την ελλειπτική καμπύλη σε κάποιο σημείο όπως προηγουμένως. Για το λόγο αυτό οι ομάδες ελλειπτικών καμπυλών περιέχουν το σημείο στο *άπειρο* $O$.Εξ ορισμού ισχύει ότι $P + (-P) = O$.

Για την πρόσθεση ενός σημείου $P$ με τον εαυτό του σχεδιάζεται μια εφαπτόμενη γραμμή στην καμπύλη από το σημείο αυτό.Εφόσον η $y$ συντεταγμένη του $P$ δεν είναι μηδέν η εφαπτόμενη τέμνει την καμπύλη σε ακριβώς ένα σημείο.Το συμμετρικό του σημείου αυτού ως προς τον άξονα $x$ είναι το αποτέλεσμα του διπλασιασμού ενός σημείου.Ισχύει δηλαδή $P + P = 2P = R$.

Σε περίπτωση που η συντεταγμένη $y$ του σημείου $P$ είναι μηδέν η εφαπτόμενη γραμμή στην καμπύλη είναι πάντα κατακόρυφη και δεν την τέμνει σε κανένα σημείο.Εξ ορισμού $2P = O$ για κάθε τέτοιο σημείο $P$.Αν κάποιος επιθυμούσε




Σχήμα 3.3: Διπλασιασμός του σημείου $P$ μιας ελλειπτικής καμπύλης.

να υπολογίσει το $3P$ θα το έκανε ως εξής: $3P = 2P + P = O + P = P$.Οπότε ισχύει ότι $3P = P$.

### Πρόσθεση Σημείων μιας Ελλειπτικής Καμπύλης (Αλγεβρική Προσέγγιση)

Αν και οι προηγούμενες γεωμετρικές περιγραφές παρέχουν μία εξαιρετική μέθοδο απεικόνισης της αριθμητικής με ελλειπτικές καμπύλες δεν είναι πρακτικές για αριθμητικούς υπολογισμούς και για αυτό χρησιμοποιούνται αλγεβρικές φόρμουλες.

Έστω $P = (x_P, y_P)$ και $Q = (x_Q, y_Q)$ ξεχωριστά σημεία μιας ελλειπτικής καμπύλης.Η πρόσθεση των δύο αυτών σημείων ορίζεται ως $P + Q = R$ όπου:

$$x_R = s^2 - x_P - x_Q \ ,$$
$$y_R = -y_P + s(x_P - x_R) \text{ και}$$
$$s = \frac{y_P - y_Q}{x_P - x_Q}$$

Αντίστοιχα για τον διπλασιασμό ενός σημείου $P = (x_P, y_P)$ εφόσον $y_P \neq 0$, έχουμε $2P = R$ όπου :

$$x_R = s^2 - 2x_P \ ,$$
$$y_R = -y_P + s(x_P - x_R) \text{ και}$$
$$s = \frac{(3x_P^2+a)}{2y_P}$$

Θυμίζουμε ότι $a$ είναι μία από τις παραμέτρους της ελλειπτικής καμπύλης.

### 3.1.2 Ομάδες Ελλειπτικών Καμπυλών Ορισμένες Πάνω από τα Πρώτα Πεδία $F_p$

Οι υπολογισμοί πάνω από το πεδίο των πραγματικών αριθμών είναι αργοί και μη ακριβείς λόγω των σφαλμάτων στρογγυλοποίησης.Οι κρυπτογραφικές εφαρμογές απαιτούν γρήγορη και ακριβή αριθμητική έτσι στην πράξη χρησιμοποιούνται οι ομάδες ελλειπτικών καμπυλών πάνω από τα δυαδικά πεδία $F_{2^m}$ καθώς και πάνω από τα πρώτα πεδία $F_p$.

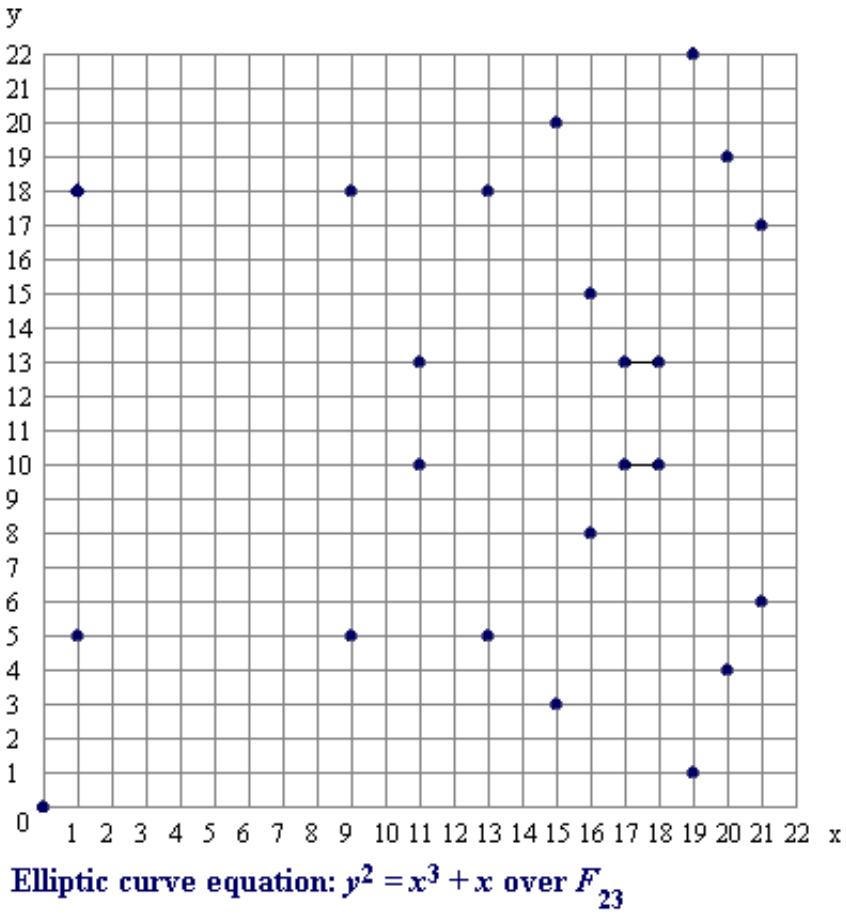
Το πεδίο των πρώτων αριθμών $F_p$ χρησιμοποιεί τους αριθμούς από 0 εως $p-1$ και όλοι οι υπολογισμοί τελειώνουν παίρνοντας το υπόλοιπο της διαίρεσης με το $p$.Για παράδειγμα, το πεδίο $F_{23}$ αποτελείται από αριθμούς ανάμεσα στο 0 και το 22 και κάθε πράξη στο πεδίο αυτό θα αποτελέσει σε έναν ακέραιο αριθμό ανάμεσα στο 0 και στο 22.Μία ελλειπτική καμπύλη με το υποκείμενο πεδίο $F_p$ δημιουργείται διαλέγοντας τις παραμέτρους $a, b$ μέσα στο πεδίο αυτό.Η καμπύλη αυτή περιλαμβάνει όλα εκείνα τα σημεία $(x, y)$ που ικανοποιούν την εξίσωση $y^2(modp) = x^3 + ax + b(modp)$.Εφόσον το $x^3 + ax + b$ δεν περιέχει επαναλαμβανόμενους παράγοντες ή ισοδύναμα ισχύει ότι $4a^3 + 27b^2 \neq 0$ τότε η εξίσωση $y^2 = x^3 + ax + b$ μπορεί να χρησιμοποιηθεί για να δημιουργήσει μία ομάδα.Μία ομάδα πάνω από τα πρώτα πεδία αποτελείται από τα σημεία της αντίστοιχης καμπύλης μαζί με ένα ειδικό σημείο $O$ που ονομάζεται σημείο στο *άπειρο*.Υπάρχουν πεπερασμένα πολλά σημεία σε μία τέτοια ελλειπτική καμπύλη.

Για παράδειγμα ας θεωρήσουμε μια ελλειπτική καμπύλη πάνω από το πρώτο πεδίο $F_{23}$.Με $a = 1$ και $b = 0$ έχουμε την καμπύλη με εξίσωση $y^2 = x^3 + x$.Το σημείο $(9, 5)$ ικανοποιεί την εξίσωση αυτή καθώς:

$$y^2(modp) = x^3 + x(modp) \rightarrow 25(mod23) = 729 + 9(mod23) \rightarrow$$
$$25(mod23) = 738(mod23) \rightarrow 2 = 2$$

Υπάρχουν 23 σημεία που ικανοποιούν την παραπάνω εξίσωση.Η γραφική παράσταση των σημείων αυτών φαίνεται παρακάτω:

$(0, 0)$ $(1, 5)$ $(1, 18)$ $(9, 5)$ $(9, 18)$ $(11, 10)$ $(11, 13)$ $(13, 5)$
$(13, 18)$ $(15, 3)$ $(15, 20)$ $(16, 8)$ $(16, 15)$ $(17, 10)$ $(17, 13)$ $(18, 10)$
$(18, 13)$ $(19, 1)$ $(19, 22)$ $(20, 4)$ $(20, 19)$ $(21, 6)$ $(21, 17)$.




Σχήμα 3.4: Τα σημεία της ελλειπτικής καμπύλης με εξίσωση $y^2 = x^3 + x$ πάνω από το πεδίο $F_{23}$.

Στο παραπάνω γράφημα παρατηρούμε ότι υπάρχουν 2 σημεία για κάθε τιμή $x$.Αν και το γράφημα φαίνεται τυχαίο υπάρχει συμμετρία γύρω από το $y = 11.5$.Πάνω από το πεδίο $F_{23}$ τα αρνητικά στοιχεία στις $y$ τιμές υπολογίζονται $modulo 23$ αποτελώντας σε θετικές αριθμούς διαφοράς από το 23.Έτσι στην προκειμένη περίπτωση $-P = (x_p, (-y_p mod 23))$.

Υπάρχουν μερικές διαφορές ανάμεσα στις ελλειπτικές καμπύλες ορισμένες πάνω από τα πρώτα πεδία με αυτές που είναι ορισμένες πάνω από τους πραγματικούς αριθμούς.Οι ομάδες ελλειπτικών καμπυλών πάνω από τα πρώτα πεδία έχουν ένα πεπερασμένο αριθμό σημείων γεγονός το οποίο είναι επιθυμητό για κρυπτογραφικούς σκοπούς.Οι καμπύλες αυτές αποτελούνται από διακριτά σημεία αλλά δεν είναι ξεκάθαρο πως γεωμετρικές σχέσεις μπορούν να εφαρμοστούν.Σαν αποτέλεσμα η γεωμετρία που αναφέρθηκε για τις ελλειπτικές καμπύλες

πάνω απο το πεδίο των πραγματικών αριθμών δεν μπορεί να χρησιμοποιηθεί και για τις καμπύλες πάνω απο τα πρώτα πεδία.Παρόλα αυτά οι αλγεβρικοί κανόνες μπορούν να προσαρμοστούν έτσι ώστε να ισχύουν και για τις ελλειπτικές καμπύλες ορισμένες πάνω από τα πρώτα πεδία.Αναφέρουμε ότι οι υπολογισμοί πάνω από τα πρώτα πεδία δεν περιέχουν σφάλματα στρογγυλοποίησης-μία βασική προυπόθεση ενός κρυπτοσυστήματος.

#### Πρόσθεση των Διακριτών Σημείων P και Q

Το αρνητικό ενός σημείου $P = (x_p, y_p)$ είναι το σημείο $-P = (x_p, -y_p mod p)$.Θεωρώντας ότι τα $P$ και $Q$ είναι διακριτά σημεία έτσι ώστε $P \neq -Q$, έχουμε $P + Q = R$ όπου :

$$x_R = s^2 - x_P - x_Q (mod p) \ ,$$
$$y_R = -y_P + s(x_P - x_R)(mod p) \text{ και}$$
$$s = \frac{y_P - y_Q}{x_P - x_Q}(mod p)$$

Υπενθυμίζουμε ότι το $a$ είναι μια από τις παραμέτρους της ελλειπτικής καμπύλης και το $s$ είναι η κλίση της ευθείας που περνάει από τα σημεία $P, Q$.

#### Διπλασιάζοντας το Σημείο P

Για τον διπλασιασμό ενός σημείου $P = (x_P, y_P)$ εφόσον $y_P \neq 0$, έχουμε $2P = R$ όπου:

$$x_R = s^2 - 2x_P (mod p) \ ,$$
$$y_R = -y_P + s(x_P - x_R)(mod p) \text{ και}$$
$$s = \frac{(3x_P^2 + a)}{2y_P}(mod p)$$

Θυμίζουμε ότι $a$ είναι μία από τις παραμέτρους της ελλειπτικής καμπύλης.

### 3.1.3 Ομάδες Ελλειπτικών Καμπυλών Ορισμένες Πάνω από τα Δυαδικά Πεδία $F_{2^m}$

Τα στοιχεία των δυαδικών πεδίων είναι m-bit αλφαριθμητικά.Οι κανόνες για την αριθμητική πάνω στα $F_{2^m}$ μπορούν να οριστούν είτε με πολυωνυμική αναπαράσταση είτε με αναπαράσταση βέλτιστης κανονικής βάσης.Επειδή τα δυαδικά πεδία $F_{2^m}$ εκτελούν πράξεις πάνω σε αλφαριθμητικά οι υπολογιστές μπορούν να εκτελέσουν αποτελεσματικά αριθμητικές πράξεις πάνω σε αυτά.

Μία ελλειπτική καμπύλη ορισμένη πάνω από ένα δυαδικό πεδίο δημιουργείται επιλέγοντας τις παραμέτρους $a$ και $b$ μέσα στο $F_{2^m}$ (η μόνη προυπόθεση είναι ότι $b \neq 0$).Έτσι η εξίσωση της ελλειπτικής καμπύλης προσαρμόζεται για δυαδική

αναπαράσταση στον εξής τύπο: $y^2+xy = x^3+ax^2+b$.Η καμπύλη αυτή περιέχει όλα τα σημεία $(x, y)$ που ικανοποιούν την παραπάνω εξίσωση πάνω από το $F_{2^m}$ (αρκεί τα $x, y$ να είναι μέλη του $F_{2^m}$).Μία ομάδα ελλειπτικής καμπύλης πάνω από τα δυαδικά πεδία αποτελείται από τα σημεία της αντίστοιχης καμπύλης μαζί με ένα ειδικό σημείο $O$ που ονομάζεται σημείο στο *άπειρο*.Υπάρχουν πεπερασμένα πολλά σημεία σε μία τέτοια ελλειπτική καμπύλη.

Ας εξετάσουμε ώς ένα μικρό παράδειγμα το δυαδικό πεδίο $F_{2^4}$ ορισμένο με πολυωνυμική αναπαράσταση με το αμείωτο πολυώνυμο $f(x) = x^4 + x + 1$.Το στοιχείο $g = (0010)$ είναι ένας γεννήτορας του πεδίου.Οι δυνάμεις του $g$ είναι:

$g^0 = (0001)$ $g^1 = (0010)$ $g^2 = (0100)$ $g^3 = (1000)$ $g^4 = (0011)$ $g^5 = (0110)$

$g^6 = (1100)$ $g^7 = (1011)$ $g^8 = (0101)$ $g^9 = (1010)$ $g^{10} = (0111)$ $g^{11} = (1110)$

$g^{12} = (1111)$ $g^{13} = (1101)$ $g^{14} = (1001)$ $g^{15} = (0001)$

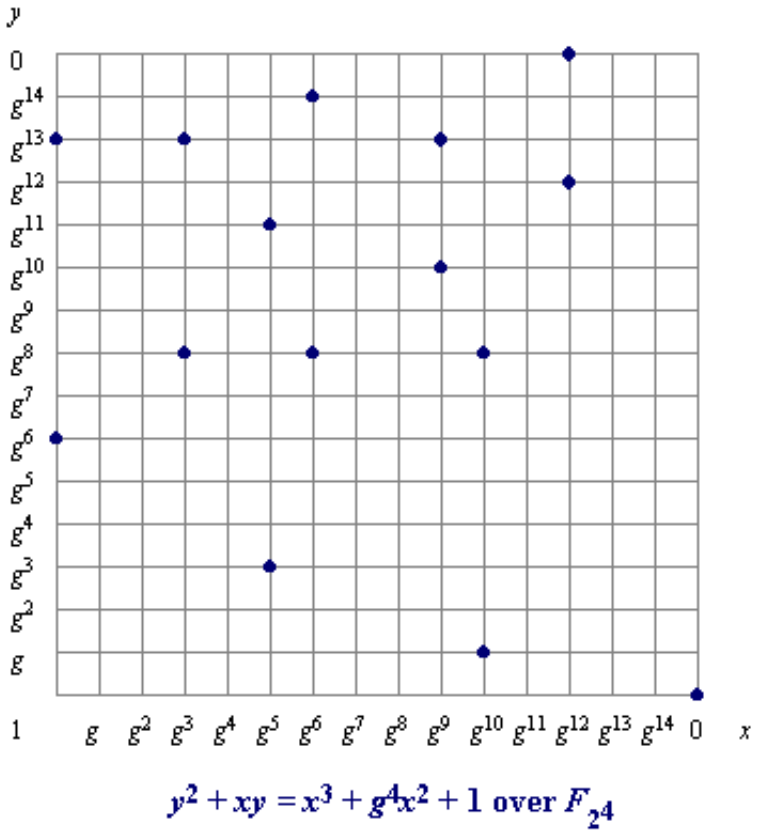
Σε μία πραγματική κρυπτογραφική εφαρμογή, η παράμετρος $m$ πρέπει να είναι αρκετά μεγάλη ώστε να αποκλείει την αποτελεσματική παραγωγή ενός τέτοιου πίνακα αλλιώς το κρυπτοσύστημα μπορεί να σπάσει.Πρακτικά, σήμερα το $m = 160$ θεωρείται κατάλληλη επιλογή.

Ας θεωρήσουμε την ελλειπτική καμπύλη με εξίσωση $y^2 + xy = x^3 + g^4x^2 + 1$.Στην περίπτωση αυτή έχουμε $a = g^4$ και $b = g^0 = 1$.Το σημείο $(g^5, g^3)$ ικανοποιεί την εξίσωση πάνω από το δυαδικό πεδίο διότι:

$y^2 + xy = x^3 + g^4x^2 + 1 \rightarrow (g^3)^2 + g^5g^3 = (g^5)^3 + g^4g^{10} + 1 \rightarrow g^6 + g^8 = g^{15}+g^{14}+1 \rightarrow (1100)+(0101) = (0001)+(1001)+(0001) \rightarrow (1001) = (1001)$.

Τα 15 σημεία που ικανοποιούν την εξίσωση φαίνονται παρακάτω:

$(1, g^{13})$ $(g^3, g^{13})$ $(g^5, g^{11})$ $(g^6, g^{14})$ $(g^9, g^{13})$
$(g^{10}, g^8)$ $(g^{12}, g^{12})$ $(1, g^6)$ $(g^3, g^8)$ $(g^5, g^3)$
$(g^6, g^8)$ $(g^9, g^{10})$ $(g^{10}, g)$ $(g^{12}, 0)$ $(0, 1)$.




Σχήμα 3.5: Τα σημεία της εξίσωσης $y^2 + xy = x^3 + g^4x^2 + 1$ ορισμένης πάνω από το δυαδικό πεδίο $F_{2^4}$.

### Πρόσθεση των Διακριτών Σημείων P και Q

Το αρνητικό του σημείου $P(x_p, y_p)$ είναι το σημείο $-P(x_p, x_p+y_p)$.Εάν τα $P, Q$ είναι διακριτά σημεία έτσι ώστε $P \neq -Q$, έχουμε $P + Q = R$ όπου:

$$x_r = s^2 + s + x_p + x_q + a,$$
$$y_r = s(x_p + x_r) + x_r + y_p \text{ και}$$
$$s = \frac{y_p - y_q}{x_p + x_q}$$

Όπως και στις ελλειπτικές καμπύλες πάνω από το πεδίο των πραγματικών αριθμών, έτσι κι εδώ ισχύει ότι $P + (-P) = O$ καθώς και $P + O = P$ για κάθε σημείο $P$ της καμπύλης.

**Διπλασιάζοντας το Σημείο P**

Εάν $x_p = 0$ τότε $2P = 0$.Δεδομένου ότι $x_p \neq 0$ έχουμε ότι $2P = R$ με:

$$x_r = s^2 + s + a \ ,$$
$$y_r = x_p^2 + (s+1)x_r \text{ και}$$
$$s = \frac{x_p + y_p}{x_p}$$

Υπενθυμίζουμε ότι το $a$ είναι μία από τις παραμέτρους της ελλειπτικής καμπύλης και το $s$ είναι η κλίση της ευθείας που περνάει από τα σημεία $P, Q$.

## 3.2 Οι Ελλειπτικές Καμπύλες στην Σύγχρονη Κρυπτογραφία

### 3.2.1 Κρυπτογραφία Δημοσίου Κλειδιού και το Πρόβλημα του Διακριτού Λογαρίθμου

Η κρυπτογραφία είναι ένας πολύ σημαντικός κλάδος της επιστήμης των υπολογιστών.Κρυπτογράφηση είναι η διαδικασία μετατροπής ενός απλού μηνύματος σε μια άλλη μορφή που δεν γίνεται κατανοητή σε άτομα που δεν έχουν εξουσιοδότηση.Η αντίστροφη διαδικασία ονομάζεται αποκρυπτογράφηση.Οι δύο παραπάνω διαδικασίες γίνονται με τη χρήση κρυπτογραφικών κλειδιών.Συγκεκριμένα, στην κρυπτογραφία δημόσιου κλειδιού(public key cryptography) η κρυπτογράφηση γίνεται με ένα *δημόσιο κλειδί* το οποίο είναι γνωστό σε όλους τους χρήστες του κρυπτογραφικού συστήματος, ενώ η αποκρυπτογράφηση γίνεται με ένα *ιδιωτικό κλειδί*.Επομένως για να επιτεθεί κανείς επιτυχώς στο κρυπτοσύστημα θα πρέπει να βρει το ιδιωτικό κλειδί γνωρίζοντας μόνο το δημόσιο κλειδί.Στα κρυπτοσυστήματα ελλειπτικών καμπυλών το δημόσιο κλειδί (public key) είναι ένα σημείο πάνω στην καμπύλη και το ιδιωτικό (private key) είναι ένας τυχαίος αριθμός.Το δημόσιο κλειδί δημιουργείται πολλαπλασιάζοντας το ιδιωτικό κλειδί με το αρχικό σημείο $G$ (generator) της καμπύλης.Το αρχικό σημείο, οι παράμετροι $a$,$b$ και κάποιες ακόμα σταθερές αποτελούν το πεδίο ορισμού των παραμέτρων της ECC.

Στα θεμέλια κάθε κρυπτοσυστήματος βρίσκεται ένα μαθηματικό πρόβλημα το οποίο είναι υπολογιστικά ακατόρθωτο να λυθεί.Το πρόβλημα του διακριτού λογαρίθμου είναι η βάση για την ασφάλεια πολλών κρυπτοσυστημάτων συμπεριλαμβανομένου και του *κρυπτοσυστήματος ελλειπτικών καμπυλών* (ECC: Elliptic Curve Cryptosystem).Πιο συγκεκριμένα το ECC βασίζεται στη δυσκολία που παρουσιάζει το *Πρόβλημα Διακριτού Λογαρίθμου στις Ελλειπτικές Καμπύλες*(ECDLP: Elliptic Curve Discrete Logarithm Problem). Προηγουμένως αναφερθήκαμε σε δύο γεωμετρικά ορισμένες πράξεις πάνω από συγκεκριμένες

ομάδες ελλειπτικών καμπυλών, την πρόσθεση σημείων και το διπλασιασμό σημείου.Διαλέγοντας ένα σημείο $P$ μιας ομάδας ελλειπτικών καμπυλών, κάποιος μπορεί να το διπλασιάσει και να υπολογίσει το σημείο $2P$.Εν συνεχεία μπορεί να προσθέσει το σημείο $P$ στο σημείο $2P$ και να υπολογίσει το σημείο $3P$.Ο καθορισμός ενός σημείου $nP$ με τον τρόπο αυτό είναι γνωστός ως *Βαθμωτός Πολλαπλασιασμός* ενός σημείου μιας ελλειπτικής καμπύλης.Το ECDLP βασίζεται στην δυσκολία αποτελεσματικής αντιστροφής του βαθμωτού πολλαπλασιασμού ενός σημείου μιας ελλειπτικής καμπύλης.

Συνήθως χρησιμοποιείται προσθετικός συμβολισμός για την περιγραφή μιας ομάδας ελλειπτικών καμπυλών.Ο βαθμωτός πολλαπλασιασμός ενός σημείου μιας ελλειπτικής καμπύλης με έναν ακέραιο αριθμό με τον προσθετικό συμβολισμό είναι ως εξής: για τον υπολογισμό του $kP$ χρειάζονται $k$ προσθέσεις του σημείου $P$ με τον εαυτό του.Χρησιμοποιώντας τον πολλαπλασιαστικό συμβολισμό η πράξη αυτή αποτελείται απο $k$ πολλαπλασιασμούς του σημείου $P$ με τον εαυτό του: $kP = P * P * P * .... * P$.Ο πιο απλός αλγόριθμος υπολογισμού του πολλαπλασιασμού ενός σημείου μιας ελλειπτικής καμπύλης με έναν ακέραιο $k$ βασίζεται στην δυαδική αναπαράσταση του $k$ και έχει πολυπλοκότητα $O(\log_2 k)$.Ο αλγόριθμος φαίνεται παρακάτω:

---

**Αλγόριθμος 6** 'Μέθοδος υπολογισμού του πολλαπλασιασμού ενός σημείου μιας ελλειπτικής καμπύλης με έναν ακέραιο'

---

**Είσοδος**: Ένα σημείο $P$ της ελλειπτικής καμπύλης και ένας ακέραιος $k$.
**Έξοδος**: Το σημείο $Q = kP$.

$Q = O$
**while** $k > 0$ **do**
  **if** $(k(mod2) == 1)$ **then**
    $Q = Q + P$
  **end if**
  $P = P + P$
  $k = k/2$
**end while**

---

Το πρόβλημα του διακριτού λογαρίθμου πάνω από πολλαπλασιαστικές ομάδες $Z_{p^*}$ είναι το εξής: δεδομένων δύο στοιχείων $r, q$ της ομάδας και ενός πρώτου αριθμού $p$ να βρεθεί ένας αριθμός $k$ έτσι ώστε $r = qk(modp)$.Ανεξαρτήτως συμβολισμού της ομάδας ελλειπτικής καμπύλης το ECDLP ορίζεται ως εξής: δεδομένων των σημείων $P, Q$ της ελλειπτικής καμπύλης να βρεθεί ένας αριθμός $k$ έτσι ώστε $Q = kP$.Το $k$ ονομάζεται ο διακριτός λογάριθμος του $Q$ στην βάση $P$.

Παράδειγμα:

Ας θεωρήσουμε την ελλειπτική καμπύλη με εξίσωση $y^2 = x^3 + 9x + 17$ ορισμένη στο πεδίο $F_{23}$.

Ψάχνουμε να βρούμε ποιός είναι ο διακριτός λογάριθμος $k$ του σημείου $Q = (4, 5)$ στην βάση $P = (16, 5)$.

Ένας απλοικός τρόπος να βρεθεί το $k$ είναι να υπολογίσουμε τα πολλαπλάσια του $P$ μέχρι να βρεθεί το $Q$.Τα πρώτα πολλαπλάσια του $P$ είναι:

$P = (16, 5)$ $2P = (20, 20)$ $3P = (14, 14)$ $4P = (19, 20)$ $5P = (13, 10)$ $6P = (7, 3)$ $7P = (8, 7)$ $8P = (12, 17)$ $9P = (4, 5)$

Παρατηρούμε ότι το $9P = (4, 5) = Q$ άρα ο διακριτός λογάριθμος του $Q$ στη βάση $P$ είναι $k = 9$.

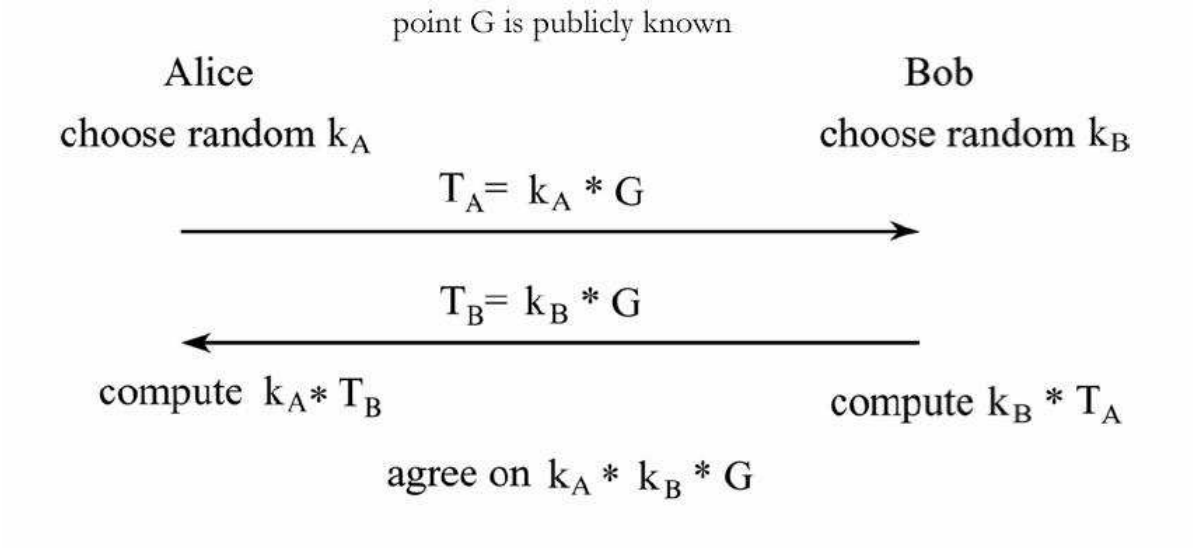
Σε μία πραγματική εφαρμογή το $k$ θα είναι τόσο μεγάλο ώστε να καθιστά αδύνατο τον υπολογισμό του με αυτόν τον τρόπο.

Όπως αναφέρθηκε και προηγουμένως στα κρυπτογραφικά συστήματα ελλειπτικών καμπυλών το δημόσιο κλειδί είναι το σημείο $Q$ και το ιδιωτικό είναι ο ακέραιος $k$.Λόγω της άμεσης αντιστοιχίας που υπάρχει μεταξύ των προβλημάτων DLP και ECDLP βασικά κρυπτογραφικά πρωτόκολλα που βασίζονται στο DLP μπορούν να εφαρμοστούν και σε συστήματα ελλειπτικών καμπυλών(π.χ. το πρωτόκολλο ανταλλαγής κλειδιών Diffie-Hellman και η κρυπτογράφηση El Gamal).Η δυσκολία ωστόσο που υπάρχει στα συστήματα ελλειπτικών καμπυλών είναι ότι πρέπει να γίνουν επιλογές ως προς τον τύπο της ελλειπτικής καμπύλης που θα χρησιμοποιηθεί, τον τύπο της πεπερασμένης ομάδας στο οποίο θα οριστεί και τους αλγορίθμους που θα χρησιμοποιηθούν για τις βασικές πράξεις.Ένας διαφορετικός συνδυασμός αυτών των επιλογών θα δώσει και διαφορετική απόδοση στο κρυπτοσύστημα.

### 3.2.2 Διάφορα Πρωτόκολλα με Χρήση Ελλειπτικών Καμπυλών

Το πρωτόκολλο Diffie-Hellman είναι το πιο γνωστό κρυπτοσύστημα ανταλλαγής κλειδιών.Το ECDH: Elliptic Curve Diffie-Hellman είναι το ανάλογο πρωτόκολλο ανταλλαγής κλειδιών με χρήση ελλειπτικών καμπυλών.Κάθε μέλος που παίρνει μέρος σε αυτό κατέχει ένα δημόσιο και ένα ιδιωτικό κλειδί.Αρχικά, γίνεται ανταλλαγή των δημοσίων κλειδιών και με βάση αυτά δημιουργείται το κοινό διαμοιραζόμενο μυστικό.Συγκεκριμένα, κάθε μέλος όταν λαμβάνει το δημόσιο κλειδί του άλλου, το πολλαπλασιάζει με το ιδιωτικό του καταλήγοντας έτσι στο κοινό μυστικό.Για παράδειγμα ας θεωρήσουμε την Alice και τον Bob που επιθυμούν να επικοινωνήσουν.Τα δύο αυτά μέλη έχουν προ-συμφωνήσει σε μια ελλειπτική καμπύλη $E$ και σε ένα σημείο γεννήτορα $G$ πάνω σε αυτήν.Αρχικά, η Alice γεννά ένα ιδιωτικό κλειδί $k_A$ που είναι ένας τυχαίος ακέραιος αριθμός και κατόπιν υπολογίζει το δημόσιο κλειδί της πολλαπλασιάζοντας το ιδιωτικό

με το σημείο $G$ της καμπύλης.Στέλνει το δημόσιο της κλειδί $k_AG$ στον Bob ο οποίος έχει υπολογίσει το ιδιωτικό του κλειδί $k_B$ και το δημόσιο του $k_BG$.Ο Bob στέλνει το δημοσιό του κλειδί στην Alice.Ο καθένας πολλαπλασιάζει το ιδιωτικό του κλειδί με το δημόσιο που έλαβε από τον άλλο καταλήγοντας στο κοινό μυστικό $k_Ak_BG$.Το πρωτόκολλο αυτό φαίνεται και στο παρακάτω σχήμα.




Σχήμα 3.6: Το πρωτόκολλο Elliptic Curve Diffie Hellman.

Το απλούστερο πρωτόκολλο κρυπτογράφησης που μπορεί να υλοποιηθεί με χρήση ελλειπτικών καμπυλών είναι το ακόλουθο.Αρχικά, το πρωτόκολλο προϋποθέτει ότι οι δύο χρήστες που επιθυμούν να επικοινωνήσουν, διαμοιράζονται το ίδιο κλειδί.Αυτό μπορεί να γίνει για παράδειγμα με χρήση του πρωτοκόλλου ανταλλαγής κλειδιού των Diffie-Hellman.Έστω ότι το κλειδί που διαμοιράζονται οι δύο χρήστες είναι το $S$.Στην περίπτωση των κρυπτογραφικών συστημάτων ελλειπτικών καμπυλών αυτό είναι ένα σημείο πάνω στην ελλειπτική καμπύλη.Αυτό που χρειάζονται και οι δύο χρήστες είναι η συντεταγμένη $x$ του σημείου $S$.Η συντεταγμένη αυτή μπαίνει ως είσοδος σε μια συνάρτηση η οποία επιστρέφει μια μάσκα(mask) που έχει μέγεθος σε bytes όσο και το μήνυμα που θέλει ο κάθε χρήστης να κρυπτογραφήσει.Για να κρυπτογραφηθεί το μήνυμα, εκτελείται η λειτουργία XOR μεταξύ του μηνύματος και της μάσκας.Για την αποκρυπτογράφηση χρησιμοποιείται η ίδια διαδικασία, αλλά τώρα γίνεται XOR το περιεχόμενο του κρυπτογραφημένου μηνύματος με την μάσκα.Για τη δημιουργία της μάσκας χρειάζεται μια συνάρτηση κατακερματισμού.

Ένα δεύτερο πρωτόκολλο που χρησιμοποιείται συχνά σε κρυπτοσυστήματα ελλειπτικών καμπυλών είναι το ECES (Elliptic Curve Encryption Scheme) και

βασίζεται στον αλγόριθμο κρυπτογράφησης El Gamal.Η κρυπτογράφηση στο πρωτόκολλο αυτό γίνεται από μία μαθηματική συνάρτηση και για το λόγο αυτό τα αρχικά μηνύματα πρέπει να αναπαρασταθούν ως μαθηματικά αντικείμενα.Η κρυπτογράφηση ενός μηνύματος που επιθυμεί να στείλει ο χρήστης Α σε ένα χρήστη Β γίνεται ως εξής:

1. Επιλέγεται ένας αριθμός $0 < k < n - 1$ όπου $n$ είναι ο μεγαλύτερος πρώτος παράγοντας της τάξης $m$ της ελλειπτικής καμπύλης.

2. Ο χρήστης Α υπολογίζει το σημείο $R = kP$, όπου $P$ είναι το σημείο βάσης με το οποίο κατασκευάστηκε και το δημόσιο κλειδί του χρήστη Β.Δηλαδή, $Q_B = k_B P$, όπου $Q_B$ είναι το δημόσιο κλειδί του Β και $k_B$ το ιδιωτικό του.

3. Υπολογίζεται το σημείο $Q = kQ_B$ (κοινό διαμοιραζόμενο μυστικό).

4. Υπολογίζεται ο ακέραιος $c = zx(modp)$, όπου $z$ είναι η αναπαράσταση του μηνύματος που κρυπτογραφείται σε μορφή ακεραίου αριθμού, $x$ είναι η x συντεταγμένη του σημείου $Q$ και $p$ είναι η τάξη της πεπερασμένης ομάδας πάνω από την οποία ορίζεται η ελλειπτική καμπύλη.

Το ζευγάρι $(R, c)$ αποτελεί την κρυπτογραφημένη μορφή του αρχικού μηνύματος.Τα βήματα της αποκρυπτογράφησης που ακολουθεί ο χρήστης Β αναφέρονται παρακάτω:

1. Υπολογίζεται το σημείο $Q = k_B R$, όπου $k_B$ είναι το ιδιωτικό κλειδί του Β.

2. Υπολογίζεται ο ακέραιος $z = \frac{c}{x}(modp)$, όπου $x$ είναι η x συντεταγμένη του σημείου $Q$ και $p$ είναι η τάξη της πεπερασμένης ομάδας πάνω από την οποία ορίζεται η ελλειπτική καμπύλη.

Ο ακέραιος $z$ είναι η αναπαράσταση του αρχικού μηνύματος σε μορφή ακεραίου αριθμού και ακολουθώντας την αντίστροφη διαδικασία με την οποία τα μηνύματα μετατρέπονται σε ακέραιους, προκύπτει το αρχικό μήνυμα που επιθυμούσε ο Α να στείλει στον Β.

Οι ελλειπτικές καμπύλες μπορούν να χρησιμοποιηθούν επίσης για τη δημιουργία ψηφιακών υπογραφών στα μηνύματα.Τα σχήματα ψηφιακών υπογραφών σχεδιάζονται για να παρέχουν ένα ψηφιακό ισοδύναμο των πραγματικών υπογραφών.Μια ψηφιακή υπογραφή είναι ένας αριθμός που εξαρτάται από το μυστικό κλειδί του ατόμου που υπογράφει επιπρόσθετα από το μήνυμα το οποίο υπογράφει.Οι ψηφιακές υπογραφές πρέπει να είναι επαληθεύσιμες, δηλαδή, αν κάποια στιγμή υπάρξει αμφιβολία για την προέλευση της υπογραφής, να φαίνεται

από ποιον πραγματικά προήλθε.Για παράδειγμα μπορεί κάποιος να πλαστογραφήσει την υπογραφή ενός άλλου ή μπορεί να υπογράψει ένα μήνυμα και μετά να αρνηθεί ότι το έκανε.Με την επαλήθευση της υπογραφής τα προβλήματα αυτά λύνονται.Ο γνωστότερος αλγόριθμος δημιουργίας ψηφιακών υπογραφών σε κρυπτογραφικά συστήματα ελλειπτικών καμπυλών είναι ο ECDSA του οποίου τα βήματα περιγράφονται παρακάτω.

1. Επιλέγεται ένας τυχαίος αριθμός $k$ με $0 \leq k \leq n-1$ όπου $n$ είναι ο μεγαλύτερος πρώτος παράγοντας της τάξης $m$ της ελλειπτικής καμπύλης.

2. Υπολογίζεται το γινόμενο $kG = (x_1, y_1)$ όπου $G$ είναι το σημείο βάσης της ελλειπτικής καμπύλης.Εν συνεχεία υπολογίζεται το $r = x_1(modn)$.Αν $r = 0$ γίνεται επιστροφή στο πρώτο βήμα.

3. Υπολογίζεται η τιμή $k^{-1}(modn)$.

4. Εφαρμόζεται μία συνάρτηση κατακερματισμού πάνω στο μήνυμα που πρόκειται να υπογραφεί.Δηλαδή $e = HASH(message)$.

5. Υπολογίζεται το $s = k^{-1}(e + dr)(modn)$ όπου $d$ είναι το ιδιωτικό κλειδί του υπογράφοντα.Αν $s = 0$ γίνεται επιστροφή στο πρώτο βήμα.

6. Επιστρέφεται το ζευγάρι $(r, s)$ που είναι και η ψηφιακή υπογραφή.

Τα βήματα επικύρωσης της υπογραφής $(r, s)$ είναι τα εξής:

1. Υπολογίζεται το $e = HASH(message)$ όπως έγινε και στο κομμάτι της δημιουργίας της υπογραφής.

2. Υπολογίζεται το $w = s^{-1}(modn)$ .

3. Υπολογίζονται τα $u_1 = ew(modn)$ και $u_2 = rw(modn)$.

4. Υπολογίζεται το σημείο $Q = u_1G + u_2Q$, όπου $Q$ είναι το δημόσιο κλειδί του υπογράφοντα.Αν το $Q$ ισούται με το σημείο στο άπειρο τότε η υπογραφή απορίπτεται.Αλλιώς, υπολογίζεται το $u = x_1(modn)$ όπου $x_1$ είναι η x συντεταγμένη του σημείου $Q$.

5. Η υπογραφή επικυρώνεται αν και μόνο αν $u = r$.

### 3.2.3 Πλεονεκτήματα Ελλειπτικών Καμπυλών

Το βασικό πλεονέκτημα των κρυπτοσυστημάτων που βασίζονται σε ελλειπτικές καμπύλες είναι η χρήση πολύ μικρότερου μέγεθους κλειδιού, σε σχέση με κάποιο άλλο σύστημα κρυπτογραφίας δημοσίου κλειδιού (πχ. RSA), για την παροχή ίδιου επιπέδου ασφάλειας.Το μικρότερο μέγεθος κλειδιού για ασφάλεια συγκεκριμένου επιπέδου, συνεπάγεται γρηγορότερες κρυπτογραφικές λειτουργίες, μικρότερες απαιτήσεις σε μνήμη καθώς και λιγότερη κατανάλωση ενέργειας γεγονός που καθιστά την κρυπτογραφία ελλειπτικών καμπυλών βιώσιμη στις συσκευές αισθητήρων.Για παράδειγμα, μια ελλειπτική καμπύλη ορισμένη πάνω από ένα 163-bit πεδίο προσφέρει την ίδια ασφάλεια με ένα 1024-bit RSA σύστημα.Η διαφορά γίνεται ακόμα πιο σημαντική καθώς το επίπεδο ασφάλειας αυξάνει.Μία ελλειπτική καμπύλη ορισμένη πάνω από ένα 512-bit πεδίο είναι ισοδύναμη σε ασφάλεια με ένα σύστημα 15.360-bit RSA.Το πρόσφατο πρότυπο κρυπτογράφησης AES: Advanced Encryption Standard προσφέρει καλύτερη ασφάλεια από τον πρόγονο του και τα μήκη κλειδιών που προτείνει(128,192 και 256 bits) κάνουν τα κρυπτοσυστήματα ελλειπτικών καμπυλών ακόμα πιο κατάλληλα για χρήση.

Πίνακας 3.1: Μέγεθος κλειδιών σε bits για ισοδύναμο επίπεδο ασφάλειας.

| Symmetric Key Cipher | Algorithm | ECC Key Length | RSA Key Length |
|---|---|---|---|
| 80 | SkipJack | 163 | 1024 |
| 112 | Triple DES | 224 | 2048 |
| 128 | 128-bit AES | 256 | 3072 |
| 192 | 192-bit AES | 384 | 7680 |
| 256 | 256-bit AES | 512 | 15360 |

### 3.2.4 Βασικές Επιθέσεις σε Κρυπτοσυστήματα Ελλειπτικών Καμπυλών

Τα κρυπτογραφικά συστήματα ελλειπτικών καμπυλών βασίζονται όπως προαναφέρθηκε στο πρόβλημα ECDLP.Για να μπορέσει κάποιος να βρει το ιδιωτικό κλειδί που χρησιμοποιείται στο σύστημα, θα πρέπει να επιλύσει ένα τέτοιο πρόβλημα.Οι σημαντικότερες μέθοδοι για την επίλυση αυτού του προβλήματος ή με άλλα λόγια, οι σημαντικότερες επιθέσεις είναι:

1. Η επίθεση MOV(MOV attack):Η επίθεση αυτή βασίζεται σε μια αναγωγή του προβλήματος ECDLP σε μία ελλειπτική καμπύλη ορισμένη πάνω από το πεδίο των πρώτων αριθμών $F_p$, στο πρόβλημα DLP στο σώμα $F_{p^l}$, όπου $l$ είναι ο μικρότερος ακέραιος για τον οποίο ισχύει η εξίσωση $p^l \equiv$

$1(modm)$ με $m$ την τάξη της καμπύλης.Για να είναι αποδοτική η επίθεση θα πρέπει ο $l$ να είναι ένας μικρός αριθμός.

2. Η επίθεση σε μη ομαλές καμπύλες (anomalous attack): Οι μη ομαλές καμπύλες είχαν προταθεί από την Miyaji και είναι πολύ ανθεκτικές στην επίθεση MOV.Ωστόσο, οι συγκεκριμένες καμπύλες δεν πρέπει να χρησιμοποιούνται σε κρυπτοσυστήματα ελλειπτικών καμπυλών αφού έχουν βρεθεί μέθοδοι που επιλύουν το ECDLP σε γραμμικό χρόνο.

3. Η μέθοδος ελάχιστου βήματος - γιγαντιαίου βήματος (Baby Step Giant Step): Η μέθοδος αυτή απαιτεί χώρο και χρόνο της τάξης του $O(\sqrt{m})$.

4. Με τη χρήση τυχαίων περιπάτων (random walks), ο χώρος που απαιτεί η παραπάνω μέθοδος μπορεί να μειωθεί σημαντικά ενώ ο χρόνος επίλυσης του προβλήματος να παραμείνει στο $O(\sqrt{m})$.Αυτό γίνεται με τις μεθόδους ρ-method και λ-method του Pollard.

5. Η μέθοδος των Pohlig και Hellman δεν αποτελεί έναν αλγόριθμο επίλυσης του ECDLP αλλά μια απλοποίηση που μπορεί να γίνει πάνω στο πρόβλημα.Οι Pohlig και Hellman παρατήρησαν ότι για την επίλυση του ECDLP σε μια ελλειπτική καμπύλη με τάξη $m = \prod_i m_i^{k_i}$ αρκεί να επιλυθούν πολλά ECDLP στις ελλειπτικές καμπύλες με τάξεις $m_i$.

Τονίζεται ότι καμία από τις παραπάνω επιθέσεις δεν μπορεί να είναι αποτελεσματική αν γίνει σωστή επιλογή της ελλειπτικής καμπύλης.Ο καλύτερος χρόνος επίλυσης του ECDLP είναι εκθετικός για τις μεθόδους 3 και 4, ενώ οι υπόλοιπες επιθέσεις μπορούν να αποφευχθούν με την κατάλληλη επιλογή της τάξης της καμπύλης.Οι συνθήκες που καθιστούν κατάλληλη την τάξη $m$ μιας ελλειπτικής καμπύλης είναι οι παρακάτω:

1. Η $m$ έχει ως παράγοντα έναν αρκετά μεγάλο πρώτο αριθμό(συνήθως μεγαλύτερο από $2^{160}$).

2. Η $m$ δεν είναι ίση με τον πρώτο αριθμό $p$.

3. Για κάθε $1 \leq k \leq 20$, ισχύει ότι $p^k \neq 1(modm)$.

Με την πρώτη συνθήκη αποφεύγεται η αποδοτική χρήση του αλγορίθμου των Pohlig και Hellman, η δεύτερη συνθήκη κάνει αδύνατη την εφαρμογή των επιθέσεων πάνω σε μη ομαλές καμπύλες ενώ η τρίτη είναι απαραίτητη για την αποφυγή των επιθέσεων MOV.Σύμφωνα με τις ήδη υπάρχουσες επιθέσεις και την ισχύ των σύγχρονων υπολογιστικών συστημάτων, ένα κλειδί πρέπει να έχει μέγεθος τουλάχιστον 160 bits για να θεωρείται ασφαλές.

# Κεφάλαιο 4

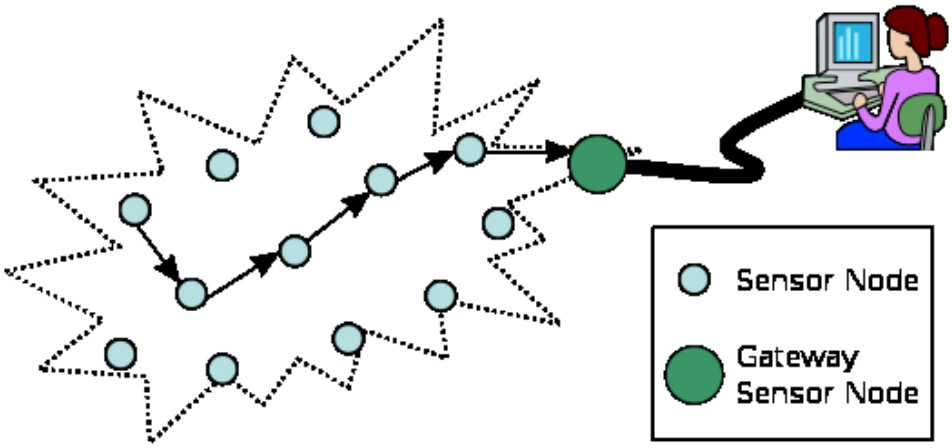
# Ασφάλεια Σε Ασύρματα Δίκτυα Αισθητήρων

## 4.1 Εισαγωγή

Ένα ασύρματο δίκτυο αισθητήρων αποτελείται από χωρικά κατανεμημένες αυτόνομες συσκευές, οι οποίες χρησιμοποιούν αισθητήρες για την συνεργατική παρακολούθηση φυσικών και περιβαλλοντικών συνθηκών.Εκτός από έναν ή περισσότερους αισθητήρες, ένας κόμβος ενός δικτύου αισθητήρων συνήθως αποτελείται από ένα ραδιο-πομποδέκτη, ένα μικροελεγκτή και μια πηγή ενέργειας, η οποία συχνά είναι μια μπαταρία.

Τα ασύρματα δίκτυα αισθητήρων προσφέρουν οικονομικά βιώσιμες λύσεις σε ποικίλες εφαρμογές.Δίκτυα τέτοιου τύπου δραστηριοποιούνται σε βιοιατρικές, στρατιωτικές, βιομηχανικές εφαρμογές καθώς και εφαρμογές παρακολούθησης του περιβάλλοντος.Τα δίκτυα αισθητήρων είναι κλειδί για τη δημιουργία *έξυπνων χώρων* που ενθέτουν τεχνολογία πληροφορίας στα καθημερινά περιβάλοντα εργασίας και κατοικίας.Παρόλα αυτά αρκετά θέματα ασφάλειας και μυστικότητας δημιουργούνται σε τέτοια δίκτυα και αποτελούν έναν πλούσιο τομέα έρευνας.Η συνεχής βελτίωση του υλικού και του λογισμικού αντιμετωπίζει αρκετά από τα προβλήματα αλλά πρέπει να δοθεί έμφαση και σε νέες υποστηριζόμενες από τις συσκευές τεχνολογίες.

Αν και η ασφάλεια δικτύων και υπολογιστικών συστημάτων είναι μια καλά εδραιωμένη επιστημονική περιοχή,με πρωτόκολλα και πρότυπα τα οποία τυγχάνουν ευρείας αναγνώρισης,η προσαρμογή και χρησιμοποίηση αυτών σε ασύρματα δίκτυα αισθητήρων, είναι τις περισσότερες φορές, αν όχι αδύνατη, πάρα πολύ δύσκολη.Αυτό συμβαίνει λόγω των ιδιαίτερων χαρακτηριστικών των δικτύων αυτών και των κόμβων που τα απαρτίζουν όπως χαμηλή υπολογιστική ισχύς, περιορισμένες ικανότητες αποθήκευσης και περιορισμένη διαθέσιμη ενέρ-




Σχήμα 4.1: Ένα τυπικό δίκτυο αισθητήρων.

γεια.Επιπλέον,κάποια ιδιαίτερα χαρακτηριστικά των διαφόρων εφαρμογών όπως λειτουργία σε αντίξοα περιβάλλοντα, ελλιπής γνώση της τοπολογίας του δικτύου, δυνατότητες αυτο-οργάνωσης και αυτόματης διόρθωσης λειτουργιών και λειτουργία χωρίς ανθρώπινη επιτήρηση καθιστούν τη διατήρηση της ασφάλειας μια μεγάλη πρόκληση.

## 4.2 Απειλές και Αρχές Ασφάλειας

### 4.2.1 Απειλές

Οι κυριότερες απειλές ενάντια στα ασφαλή δίκτυα αισθητήρων είναι:

- Μη εξουσιοδοτημένη ακρόαση - Eavesdropping: Ονομάζεται η με μη ενεργητικούς τρόπους απόκτηση πληροφοριών από ένα δίκτυο.Η φύση της επικοινωνίας στα δίκτυα αισθητήρων, η οποία βασίζεται στην χρησιμοποίηση ραδιοφωνικών συχνοτήτων για την διεξαγωγή της, την καθιστά ευπαθή σε υποκλοπές από οποιονδήποτε δέκτη βρίσκεται εντός της ακτίνας εκπομπής των κόμβων του δικτύου.Σε κάποιες περιπτώσεις υπάρχει η δυνατότητα υποκλοπών και από δέκτες που βρίσκονται έξω από την ακτίνα εκπομπής των δικτύων,αρκεί να διαθέτουν κεραίες υψηλού κέρδους.Το μεγαλύτερο πρόβλημα που προκαλείται από την απειλή αυτή σχετίζεται με την δυνατότητα που έχει ο υποκλοπέας να επεξεργάζεται τα μηνύματα που υποκλέπτει και να τα χρησιμοποιεί για την εκτόξευση επιθέσεων προς το ασύρματο δίκτυο όπως θα δούμε και παρακάτω.

- Εκπομπή Λαθεμένων ή Επανεκπομπή Ετερεχρονισμένων Παλαιότερων Μηνυμάτων - Message Injection or Replay Attack:η εισαγωγή λαθεμένων ή παλαιών μηνυμάτων στο δίκτυο απο μη-εξουσιοδοτημένους κόμβους είναι μία πάγια απειλή τόσο ως προς τους κόμβους διότι προκαλεί την ταχεία αποφόρτιση τους εξαιτίας της εντατικής χρήσης του πομποδέκτη τους όσο και ως προς την αξιοπιστία του δικτύου εξαιτίας των λαθεμένων ή ετεροχρονισμένων πληροφοριών που μεταδίδονται μέσα σε αυτό.Τέτοιου είδους τακτικές είναι τυπικές στις επιθέσεις τύπου Denial of Service.

- Αντιποίηση και Μη Εξουσιοδοτημένη Μεταβολή Μηνυμάτων - Impersonation and Message Modification: Η αντιποίηση λαμβάνει χώρα όταν ένας μη εξουσιοδοτημένος κόμβος προσποιείται τη συμπεριφορά ενός κόμβου του δικτύου.Η μη εξουσιοδοτημένη μεταβολή ενός μηνύματος συμβαίνει όταν κάποιος επιτιθέμενος καταφέρει να συλλέξει ένα μήνυμα ,να το μεταβάλει και στη συνέχεια να το επανεκπέμψει στο δίκτυο.Η διαδικασία αυτή είναι τυπική στις επιθέσεις του τύπου Man In the Middle.

- Ανάλυση Κίνησης και Εκμετάλλευση Παράπλευρων Πληροφοριών Καναλιού - Traffic - Side Channel Analysis: Ανάλυση κίνησης συμβαίνει όταν ένας επιτιθέμενος είναι σε θέση να καταγράφει όλη την κίνηση σε μια περιοχή του δικτύου.Η ανάλυση της κίνησης μπορεί να αποδειχθεί ένα πολύ σημαντικό εργαλείο στα χέρια ενός επιτιθέμενου καθώς μπορεί με κατάλληλη επεξεργασία να ανακαλύψει την τοπολογία του δικτύου,πρότυπα και συνήθειες επικοινωνίας,σχέσεις μεταξύ κόμβων του δικτύου καθώς και κόμβους του δικτύου σημαντικούς για τη λειτουργία του(π.χ. σταθμός βάσης).Η ανάλυση τύπου Side Channel βασίζεται στις *παράπλευρες* πληροφορίες που μπορούν να εξαχθούν από αυτό.Ως παράπλευρη χαρακτηρίζεται οποιαδήποτε πληροφορία ανακτάται από τον κόμβο, και η οποία δεν είναι το ίδιο το μήνυμα, όπως απαιτούμενος χρόνος για εκτέλεση κρυπτογραφικών αλγορίθμων, κατανάλωση ισχύος,μηνύματα λάθους κτλ.


Σχήμα 4.2: Απειλές Ενάντια σε Ασύρματα Δίκτυα Αισθητήρων

Για να μπορέσει ένα σύστημα να αντιμετωπίσει τις παραπάνω απειλές θα πρέπει να είναι σύμφωνο με τις αρχές ασφάλειας που παρουσιάζονται παρακάτω.

### 4.2.2 Αρχές Ασφάλειας

Οι βασικές αρχές ασφάλειας σε ασύρματα δίκτυα αισθητήρων παρουσιάζονται παρακάτω:

- **Εμπιστευτικότητα**: Οι πληροφορίες που διακινούνται σε ένα δίκτυο, θα πρέπει να αποκαλύπτονται στους αποδέκτες στους οποίους απευθύνονται και μόνο σε αυτούς.Οποιοσδήποτε άλλος δεν πρέπει, ακόμα και αν κάποιο μήνυμα καταλήξει σε αυτόν από λάθος, να είναι σε θέση να ανακτήσει τις πληροφορίες που περιέχονται σε αυτό.

- **Μη Αποποίηση**: Η μη αποποίηση συνίσταται στην ικανότητα του δικτύου να ταυτοποιεί μοναδικά οποιαδήποτε δραστηριότητα στο δίκτυο, ώστε να μπορεί να εντοπιστεί οποιαδήποτε μη επιθυμητή δραστηριότητα σε αυτό.

- **Αυθεντικότητα**: Οι πληροφορίες οι οποίες διακινούνται στο δίκτυο θα πρέπει να προέρχονται απο τα στοιχεία που το αποτελούν και από τυχόν τρίτες σαφώς καθορισμένες οντότητες.Οποιοσδήποτε άλλος δεν θα πρέπει να είναι ικανός να ανακτήσει πληροφορίες από το δίκτυο, είτε με λαθραία ακρόαση των διακινούμενων μηνυμάτων είτε μετά από καταγραφή και επεξεργασία αυτών, ή να το χρησιμοποιήσει για αποστολή δικών του μηνυμάτων.

- **Ακεραιότητα**: Το δίκτυο πρέπει να παρέχει εγγυήσεις τέτοιες ώστε να διασφαλίζεται η επικοινωνία μεταξύ των κόμβων, χωρίς να είναι εφικτή η μεταβολή του μηνύματος, κατά τη διάρκεια της δρομολόγησης του από τον αποστολέα στον παραλήπτη, από κάποιον κακόβουλο τρίτο ή από λάθος μετάδοσης.Σε περίπτωση που κάτι τέτοιο συμβεί ο αποδέκτης θα πρέπει να είναι σε θέση να εντοπίσει τη μεταβολή αυτή.

- **Διαθεσιμότητα**: Οι υπηρεσίες του δικτύου θα πρέπει να είναι διαθέσιμες προς χρήση, από οποιονδήποτε εξουσιοδοτημένο χρήστη του, οποτεδήποτε εκείνος τις χρειαστεί.

- **Ελεγχόμενη Πρόσβαση**: Η πρόσβαση στις υπηρεσίες και τους πόρους του δικτύου θα πρέπει να γίνεται με ελεγχόμενο τρόπο, έτσι ώστε κάθε στοιχείο του να έχει πρόσβαση μόνο στις υπηρεσίες και πόρους για τους οποίες έχει το δικαίωμα.

- **Φρεσκάδα Διακινούμενων Πληροφοριών**: Το δίκτυο πρέπει να εξασφαλίζει τη φρεσκάδα, ως προς το χρόνο, των πληροφοριών που διακινούνται σε αυτό.Με τον τρόπο αυτό εξασφαλίζεται η προστασία των στοιχείων του από τις επιθέσεις επανεκπομπής μηνυμάτων, καθώς επίσης και από την άσκοπη κατανάλωση ενεργειακών πόρων στους κόμβους από μηνύματα τα οποία για κάποιους λόγους κυκλοφορούν διαρκώς (ατέρμονες βρόχους) στο δίκτυο.

## 4.3 Επιθέσεις-Αντίμετρα

Σε ένα δίκτυο, ένας επιτιθέμενος είναι μια κακόβουλη οντότητα που έχει ως βασικό σκοπό να υποκλέψει τις πληροφορίες που ανταλλάσονται μέσα σε αυτό.Στα ασύρματα δίκτυα αισθητήρων, ένας επιτιθέμενος μπορεί να έχει στην κατοχή του κάποιες συσκευές αισθητήρων παρόμοιες με αυτές του δικτύου (mote-class attacker) ή ένα laptop με κεραία ασύρματης μετάδοσης (laptop-class attacker).Στη δεύτερη περίπτωση ο επιτιθέμενος έχει σημαντικό πλεονέκτημα έναντι των έντιμων κόμβων του δικτύου διότι ένα laptop έχει πολύ δυνατότερη υπολογιστική ισχύ και πιθανότατα δυνατότερη κεραία μετάδοσης από τις συσκευές του δικτύου.Επίσης, οι επιθέσεις σε ασύρματα δίκτυα αισθητήρων χωρίζονται σε εσωτερικές (insider attacks) και εξωτερικές επιθέσεις (outsider attacks).Στο πρώτο είδος, ο επιτιθέμενος έχει υπο την κατοχή του κάποιες εξουσιοδοτημένες συσκευές του δικτύου (π.χ. έχει υποκλέψει τα κρυπτογραφικά κλειδιά τους) ενώ στο δεύτερο ο επιτιθέμενος δεν έχει πρόσβαση στο δίκτυο.

Παρακάτω αναφέρονται οι πιο σημαντικές επιθέσεις σε ασύρματα δίκτυα αισθητήρων και τα πιθανά αντίμετρα σε αυτές ανά επίπεδο του μοντέλου OSI(Open Systems Interconnection Reference Model) .Οι επιθέσεις αυτές στοχεύουν σε διαφορετικές λειτουργίες του δικτύου και για αυτό το λόγο τα αντίμετρα και η συνολική πολιτική ασφάλειας πρέπει να σχεδιάζονται όχι μεμονωμένα και ανεξάρτητα, αλλά με μια πολυεπίπεδη προοπτική και αρχιτεκτονική.

### 4.3.1 Επιθέσεις στο Φυσικό Επίπεδο

Οι επιθέσεις στο φυσικό επίπεδο μπορούν να στοχεύουν είτε στο μέσο μετάδοσης είτε στον ίδιο τον κόμβο ως συσκευή.Στις πρώτες, ο επιτιθέμενος στοχεύει στην παύση λειτουργίας του δικτύου, μέσω της απαγόρευσης χρήσης του RF φάσματος εκπομπής σε αυτό(jamming attack).Η απαγόρευση επιτυγχάνεται με τη χρήση ισχυρών παρεμβολών οι οποίες κατακλύζουν την περιοχή του ραδιοφωνικού φάσματος συχνοτήτων που χρησιμοποιεί το δίκτυο με *θόρυβο*, έτσι ώστε να είναι αδύνατη η ανταλλαγή οποιουδήποτε μηνύματος ανάμεσα στους κόμβους.Ένα αντίμετρο το οποίο αντιμετωπίζει ικανοποιητικά την παραπάνω

απειλή είναι η χρήση τεχνικών διαμόρφωσης εύρους φάσματος στο σήμα και στη συχνότητα για τη μετάδοση.Οι τεχνικές αυτές επιτρέπουν την επικοινωνία μόνο σε κόμβους οι οποίοι γνωρίζουν εκ των προτέρων το χρησιμοποιούμενο μοντέλο επικοινωνίας, αποκρύπτοντας την ύπαρξη και λειτουργία των κόμβων στην περιοχή ανάπτυξης τους.

Στις επιθέσεις ενάντια στους κόμβους ως συσκευές (tampering attack), ένας επιτιθέμενος εκμεταλεύεται τη λειτουργία των κόμβων σε περιοχές χωρίς ανθρώπινη επιτήρηση και στοχεύει με την ανάκτηση τους στη φυσική καταστροφή τους αλλά και στην απόκτηση τυχόν πολύτιμων δεδομένων (π.χ. κρυπτογραφικά κλειδιά) που είναι αποθηκευμένα σε αυτούς.Η κύρια προστασία έναντι σε αυτού του είδους την επίθεση είναι η χρήση μηχανισμών προστασίας ενάντια στην παραβίαση, η χρήση κόμβων όσο το δύνατον μικρότερου μεγέθους καθώς και η χρήση τεχνικών φυσικής απόκρυψης τους(χρώμα,σχήμα συσκευών).

### 4.3.2 Επιθέσεις στο Επίπεδο Ζεύξης των Δεδομένων

Στο επίπεδο ζεύξης των δεδομένων μπορούμε να εντοπίσουμε δύο είδη επιθέσεων:

- τις επιθέσεις οι οποίες προκαλούν σύγκρουση των μεταδιδόμενων πακέτων δεδομένων
- τις επιθέσεις οι οποίες στοχεύουν στην εξάντληση των αποθεμάτων των συσσωρευτών κόμβων

Οι πρώτες στοχεύουν σε δίκτυα τα οποία χρησιμοποιούν MAC πρωτόκολλα τα οποία λειτουργούν με το σχήμα Ready-to-Send/Clear-to-Send(RTS/CTS). Στα δίκτυα αυτά ο επιτιθέμενος τοποθετεί στην περιοχή λειτουργίας του δίκτου κόμβους οι οποίοι συλλαμβάνουν τα μηνύματα (RTS/CTS), υποδύονται ότι είναι οι *νόμιμοι* αποδέκτες του αιτήματος και στην περίπτωση του RTS δεν αποστέλουν ποτέ το CTS με αποτέλεσμα να εξαναγκάζεται ο κόμβος να εκπέμπει συνεχώς RTS πακέτα, καταργώντας ουσιαστικά τη δυνατότητα επικοινωνίας με το υπόλοιπο δίκτυο.Η βασικότερη μέθοδος αντιμετώπισης της παραπάνω επίθεσης, είναι η χρήση πρωτοκόλλων MAC τα οποία δεν επιτρέπουν τις συγκρούσεις πακέτων δεδομένων καθώς και η χρήση κωδίκων διόρθωσης λαθών.

Οι δεύτερες στοχεύουν στην εξάντληση των ενεργειακών αποθεμάτων των κόμβων, την οποία επιτυγχάνουν με τις συνεχείς αιτήσεις δρομολόγησης μεγάλων σε μέγεθος μηνυμάτων χρησιμοποιώντας το υποσύστημα μετάδοσης το οποίο καταναλώνει τη μεγαλύτερη ενέργεια από όλα τα υποσυστήματα ενός κόμβου.Η κύρια άμυνα σε αυτού του είδους την επίθεση είναι η δρομολόγηση των μηνυμάτων μόνο μετά την αυθεντικοποίηση του αποστολέα καθώς και η φραγή των μηνυμάτων με μέγεθος μεγαλύτερο από το μέγεθος των τυπικών μηνυμάτων της εφαρμογής που εξυπηρετεί το δίκτυο.

### 4.3.3 Επιθέσεις στα Επίπεδα Δικτύου και Μεταφοράς

Στο επίπεδο δικτύου μπορούν να εκτοξευτούν μια σειρά από επιθέσεις οι οποίες στοχεύουν στην εκμετάλλευση αδυναμιών των πρωτοκόλλων δρομολόγησης.Οι βασικές επιθέσεις αναφέρονται παρακάτω:

- Τροποποίηση ή Επανεκπομπή Πληροφοριών Δρομολόγησης: Η πιο άμεση επίθεση σε ένα πρωτόκολλο δρομολόγησης στοχεύει στην πληροφορία δρομολόγησης που ανταλάσεται μεταξύ κόμβων του δικτύου.Τροποποιώντας ή επανεκπέμποντας την πληροφορία αυτή, ένας επιτιθέμενος μπορεί να δημιουργήσει ατέρμονες βρόχους δρομολόγησης,να επεκτείνει ή να μικρύνει διαδρομές προς και από την πηγή,να προσελκύει ή να απωθεί την κίνηση του δικτύου σε περιοχές που έχει όφελος καθώς και να αυξήσει την συνολική από *άκρο σε άκρο* καθυστέρηση.

- Επιλεκτική Προώθηση Λαμβανόμενων Μηνυμάτων: Τα ασύρματα δίκτυα συνήθως βασίζονται στην υπόθεση ότι οι συμμετέχοντες κόμβοι θα προωθούν πιστά τα μηνύματα που λαμβάνουν.Στην επίθεση τύπου επιλεκτικής προώθησης ένας κακόβουλος κόμβος αρνείται να προωθήσει συγκεκριμένα μηνύματα και τα απορίπτει, εξασφαλίζοντας έτσι ότι αυτά δεν μεταδίδονται πιο πέρα μέσα στο δίκτυο.Μια απλή μορφή αυτής της επίθεσης είναι όταν ένας κακόβουλος κόμβος συμπεριφέρεται σαν μία *μαύρη τρύπα* (Black Hole Attack) απορίπτοντας όποιο πακέτο φτάνει στην κατοχή του.Βέβαια, με την επίθεση αυτή οι γείτονες του κακόβουλου κόμβου μπορεί να θεωρήσουν ότι αυτός απέτυχε και να αναζητήσουν άλλες διαδρομές.

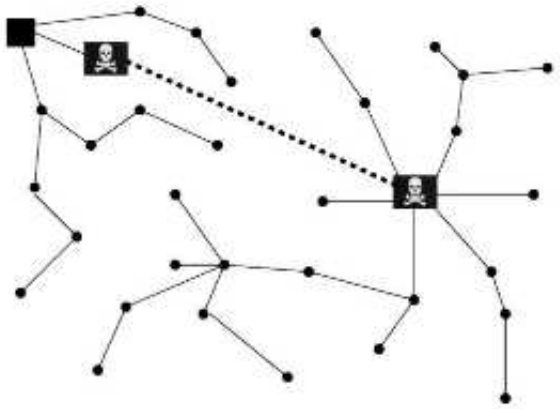
- Επίθεση τύπου Sinkhole (Sinkhole Attack): Στην επίθεση τύπου Sinkhole στόχος του επιτιθέμενου είναι να παρασύρει σχεδόν όλη την κίνηση μιας συγκεκριμένης περιοχής διαμέσου ενός κακόβουλου κόμβου.Επειδή οι κόμβοι που βρίσκονται πάνω ή κοντά στο μονοπάτι που ακολουθούν τα πακέτα έχουν τη δυνατότητα αλλοιώσουν τα δεδομένα της εφαρμογής, οι επιθέσεις τύπου Sinkhole μπορούν να ενεργοποιήσουν και άλλες επιθέσεις (π.χ. επιλεκτική προώθηση).Οι επιθέσεις τύπου Sinkhole επιτυγχάνουν τον σκοπό τους κάνοντας τον κακόβουλο κόμβο να φαίνεται ιδιαίτερα ελκυστικός προς τους άλλους κόμβους σε σχέση με τον αλγόριθμο δρομολόγησης.Για παράδειγμα, ένας επιτιθέμενος θα μπορούσε να εκπέμπει συνεχώς μια *διαφήμιση* για μια διαδρομή υψηλής ποιότητας διαμέσου ενός κακόβουλου κόμβου προς το σταθμό βάσης.Τονίζουμε ότι τα ασύρματα δίκτυα αισθητήρων είναι ιδιαίτερα ευπαθή σε αυτού του είδους επίθεση

λόγω της ειδικού μοτίβου επικοινωνίας που ακολουθούν(π.χ. όλα τα πακέτα έχουν τον ίδιο προορισμό σε ένα δίκτυο με ένα σταθμό βάσης).

- Επίθεση τύπου Sybil (Sybil Attack): Στην επίθεση τύπου Sybil ένας κακόβουλος κόμβος παρουσιάζεται με πολλές διαφορετικές ταυτότητες στους γειτονές του.Με τον τρόπο αυτό παρουσιάζονται σημαντικά προβλήματα σε σχήματα τα οποία είναι ανεκτικά στα σφάλματα, όπως η κατανεμημένη αποθήκευση, η γεωγραφική δρομολόγηση και η συντήρηση της τοπολογίας του δικτύου.

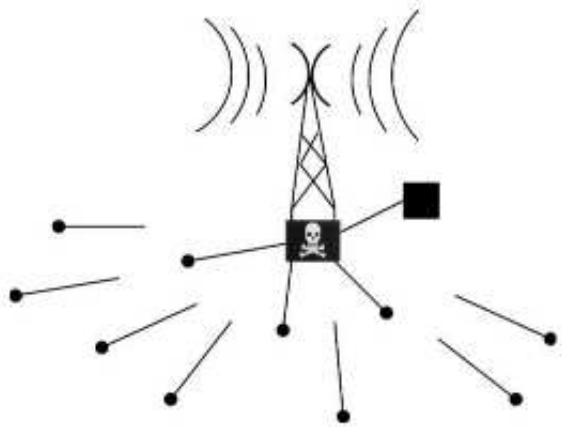
- Επίθεση τύπου Wormhole (Wormhole Attack): Σε αυτό τον τύπο επίθεσης, ένας επιτιθέμενος καθοδηγεί τα μηνύματα που έλαβε από ένα μέρος του δικτύου μέσω ενός μικρής καθυστέρησης συνδέσμου και τα επανεκπέμπει σε ένα άλλο μέρος του δικτύου.Η πιο απλή επίθεση τέτοιου είδους είναι η τοποθέτηση ενός κακόβουλου κόμβου ανάμεσα σε δύο κανονικούς, με τον κακόβουλο να προωθεί συνεχώς μηνύματα μεταξύ των δύο κόμβων.Ακόμα ένας κακόβουλος κόμβος τοποθετημένος κοντά σε ένα σταθμό βάσης μπορεί να διαταράξει τη δρομολόγηση, πείθωντας κόμβους που φυσιολογικά βρίσκονται αρκετά βήματα μακριά από το σταθμό βάσης ότι βρίσκονται ένα βήμα μακριά από αυτόν.Έτσι δημιουργείται ένα sinkhole και όλη η κίνηση στην κοντινή περιοχή θα περνάει από τον κακόβουλο κόμβο.

- Επίθεση τύπου Hello Flood: Πολλά πρωτόκολλα απαιτούν ,κατά την εκκίνηση τους, από τους κόμβους να ανακοινώσουν τους εαυτούς τους στους γείτονες τους στέλνοντας ένα HELLO μήνυμα.Έτσι ένας κόμβος που λαμβάνει ένα τέτοιο μήνυμα από κάποιον άλλο θεωρεί ότι βρίσκεται μέσα στην ακτίνα εκπομπής του αποστολέα.Αυτή η θεώρηση μπορεί να μην είναι αληθής καθώς ένας επιτιθέμενος που έχει στην κατοχή του ένα laptop μπορεί να εκπέμπει πληροφορία με τόσο ισχυρό σήμα μετάδοσης ώστε να πείσει κάθε κόμβο του δικτύου ότι είναι γείτονας του.Οι επιθέσεις τύπου Hello Flood μπορούν να θεωρηθούν ως *μιας κατεύθυνσης* μεταδιδόμενες wormholes.



Σχήμα 4.3: Η επίθεση τύπου wormhole σε ένα δίκτυο αισθητήρων.

Για την προστασία των πρωτοκόλλων δρομολόγησης σε ασύρματα δίκτυα αισθητήρων από ωτακουστές, λανθασμένη πληροφορία δρομολόγησης, επιθέσεις τύπου Sybil και επιθέσεις τύπου Hello Flood απαιτείται κρυπτογράφηση σε επίπεδο συνδέσμου και αυθεντικότητα από κόμβο σε κόμβο, δρομολόγηση μέσω πολλαπλών μονοπατιών, εξακρίβωση ταυτότητας των κόμβων καθώς και εξακρίβωση της διπλής κατεύθυνσης των συνδέσμων.Οι επιθέσεις τύπου Wormhole και Sinkhole θέτουν σημαντικές προκλήσεις στον σχεδιασμό ασφαλών πρωτοκόλλων δρομολόγησης.Δεν υπάρχουν συγκεκριμένα αντιμέτρα για τις επιθέσεις αυτές αλλά μπορούν να οριστούν μετά τον σχεδιασμό του εκάστοτε πρωτοκόλλου δρομολόγησης.Γενικότερα, τα πρωτόκολλα δρομολόγησης θα πρέπει να σχεδιάζονται έτσι ώστε να παρακολουθούν συνεχώς τη λειτουργία του δικτύου, να απομονώνουν μη φυσιολογικές συμπεριφορές, να παρέχουν εναλλακτικά μονοπάτια δρομολόγησης και να επιτρέπουν τη συμμετοχή στη δρομολόγηση μόνο σε εξουσιοδοτημένους κόμβους.



Σχήμα 4.4: Η επίθεση τύπου HELLO flood σε ένα δίκτυο αισθητήρων.

## 4.4 Κατηγοριοποίηση των Πρωτοκόλλων Ασφαλείας

Οι βασικές κατηγορίες στις οποίες διακρίνονται τα πρωτόκολλα ασφάλειας σε ασύρματα δίκτυα αισθητήρων αναφέρονται παρακάτω:

- Πρωτόκολλα που χρησιμοποιούν τρίτη έμπιστη οντότητα: Τα πρωτόκολλα της κατηγορίας αυτής απαιτούν για την ανταλλαγή μηνυμάτων την ύπαρξη μιας τρίτης έμπιστης οντότητας η οποία παίζει το ρόλο του διανομέα κλειδιών ανάμεσα στους κόμβους που επιθυμούν να επικοινωνήσουν.Επιπλέον, η οντότητα αυτή αναλαμβάνει με κάποιο τρόπο να πιστοποιεί τους δύο κόμβους που θέλουν να επικοινωνήσουν καθώς και να ενημερώνει τον καθένα για την πιστοποίηση του άλλου.

  Ένα παράδειγμα τέτοιου πρωτοκόλλου είναι το SPINS: Security Protocols for Sensor Networks.Αυτή η αρχιτεκτονική ασφάλειας που αναπτύχθηκε από τον Adrian Perrig αποτελείται από μια σειρά από πρωτόκολλα για την παροχή ασφάλειας και αυθεντικοποιημένης εκπομπής, στηριζόμενα στις αρχές της συμμετρικής κρυπτογραφίας.Η αρχιτεκτονική SPINS στηρίζεται σε δύο δομικά στοιχεία,το SNEP,ένα πρωτόκολλο το οποίο παρέχει εμπιστευτικότητα,αυθεντικοποίηση και φρεσκάδα δεδομένων, και το μTesla, το οποίο εξασφαλίζει την αυθεντικότητα των εκπομπών.Η βασική ιδέα πίσω από την αρχιτεκτονική αυτή είναι ότι κάθε κόμβος κατέχει ένα μυστικό κλειδί,την γνώση του οποίου μοιράζεται με έναν έμπιστο σταθμό βάσης ο οποίος είναι διαρκώς διαθέσιμος και ικανός να επικοινωνεί

με οποιονδήποτε κόμβο του δικτύου.Όταν δύο κόμβοι επιθυμούν να επικοινωνήσουν μεταξύ τους με ασφάλεια,ο σταθμός βάσης αναλαμβάνει να παρεμβληθεί ανάμεσα τους και να λειτουργήσει ως κέντρο διανομής κρυπτογραφικών κλειδιών.

Τα πρωτόκολλα αυτής της κατηγορίας παρουσιάζουν σημαντικά προβλήματα ιδιαίτερα στα δίκτυα με μεγάλη κάλυψη στο χώρο, με σημαντικότερο αυτό της τοποθέτησης του έμπιστου σταθμού βάσης στο χώρο έτσι ώστε να είναι προσπελάσιμος συνεχώς και απευθείας από οποιονδήποτε κόμβο του δικτύου.Ακόμα, ο ad-hoc τρόπος ανάπτυξης και κατανομής της τοπολογίας του δικτύου στο χώρο,καθιστά τα πρωτόκολλα της κατηγορίας αυτής δύσκολα στη χρήση για εφαρμογές ασυρμάτων δικτύων αισθητήρων.

- Πρωτόκολλα με Χρήση Κρυπτογραφίας Δημόσιου-Ιδιωτικού Κλειδιού: Αυτή η κατηγορία πρωτοκόλλων περιλαμβάνει εκείνα τα οποία χρησιμοποιούν για την ανταλλαγή μηνυμάτων κρυπτογραφία δημόσιου-ιδιωτικού κλειδιού.

  Ένα παράδειγμα τέτοιου πρωτοκόλλου είναι το LEAP: Localized Encryption and Authentication Protocol και παρουσιάστηκε από τον Zhu.Το πρωτόκολλο αυτό υποστηρίζει τέσσερα είδη κλειδιών σε κάθε κόμβο: ένα ατομικό κλειδί μοιραζόμενο με το σταθμό βάσης, ένα κλειδί μοιραζόμενο με έναν άλλο κόμβο,ένα κλειδί τομέα (cluster key) μοιραζόμενο με πολλαπλούς γειτονικούς κόμβους και ένα κλειδί ομάδας (group key) το οποίο κατέχουν όλοι οι κόμβοι του δικτύου.Το πρωτόκολλο αυτό αποτρέπει την εκτόξευση αρκετών επιθέσεων που είναι συνήθεις στα ασύρματα δίκτυα αισθητήρων.

  Αποδεικνύεται ότι τα πρωτόκολλα αυτή της κατηγορίας δεν είναι κατάλληλα για χρήση σε ασύρματα δίκτυα αισθητήρων εξαιτίας της μεγάλης κατανάλωσης ενέργειας για την εκτέλεση των απαιτούμενων από το πρωτόκολλο βημάτων (handshaking,ανταλλαγή κλειδιών).

- Πρωτόκολλα με Χρήση Προυπολογισμένων και Προτοποθετημένων Κρυπτογραφικών Κλειδιών: Η κατηγορία αυτή περιλαμβάνει πρωτόκολλα που χρησιμοποιούν τεχνικές συμμετρικής κρυπτογραφίας,με κλειδιά τα οποία έχουν προ-υπολογιστεί και τοποθετηθεί στους κόμβους, πριν από την ανάπτυξη του δικτύου.Οι κόμβοι μπορούν να χρησιμοποιήσουν τα κλειδιά αυτά για ασφαλή επικοινωνία αλλά και για τη δημιουργία νέων κλειδιών.

  Ένα δημοφιλές πρωτόκολλο αυτής της κατηγορίας είναι το TinySec που προτάθηκε από τον Karlof.Είναι το πρώτο πλήρως-υλοποιημένο πρωτόκολλο που υποστηρίζει κρυπτογραφία σε επίπεδο συνδέσμου σε ασύρματα

δίκτυα αισθητήρων.Το TinySec βασίζεται στην ιδέα ότι ο εντοπισμός μη-εξουσιοδοτημένων πακέτων είναι πιο εύκολος στο επίπεδο ζεύξης των δεδομένων.Με τον τρόπο αυτό, μπορεί να αποφευχθεί η άσκοπη δρομολόγηση πακέτων, εξοικονομώντας ενέργεια και bandwidth.Παρέχει μηχανισμούς αυθεντικοποίησης και ακεραιότητας των μηνυμάτων (με χρήση Message Authentication Code , CBC-MAC), εμπιστευτικότητα(μέσω κρυπτογράφησης SkipJack),σημασιολογική ασφάλεια (με χρήση Initialization Vector) και προστασία από επανάληψη μηνυμάτων.

Ακόμα μια πολύ σημαντική εργασία στον χώρο της ασφάλειας των δικτύων αισθητήρων είναι αυτή των Eschenauer και Gligor οι οποίοι ήταν οι πρώτοι που παρουσίασαν ένα σύστημα ασφαλούς λειτουργίας το οποίο βασίζεται στην ύπαρξη μιας δεξαμενής κρυπτογραφικών κλειδιών.Τα κλειδιά αυτά υπολογίζονται και τοποθετούνται στους κόμβους πριν από την ανάπτυξη τους στην περιοχή παρατήρησης, εξασφαλίζοντας την ασφαλή διανομή,ανανέωση και κατάργηση κρυπτογραφικών κλειδιών καταναλώνοντας όσο το δυνατόν λιγότερη ενέργεια εξαιτίας της μείωσης των απαιτούμενων υπολογισμών.Το μειονέκτημα της παραπάνω εργασίας είναι οι υψηλές απαιτήσεις σε μνήμη για την προαποθήκευση των κλειδιών, σε περίπτωση μεγάλου αριθμού κόμβων.

Τα χαρακτηριστικά των πρωτοκόλλων ασφάλειας που περιγράψαμε παραπάνω, συνοψίζονται στον παρακάτω πίνακα.

Πίνακας 4.1: Χαρακτηριστικά των πρωτοκόλλων ασφάλειας

| Πρωτόκολλο | Κρυπτογράφηση | Χαρακτηριστικά | Μειονεκτήματα |
|---|---|---|---|
| Perrig (SPINS) Zhu (LEAP) | συμμετρική | Απαιτεί την ύπαρξη τρίτης έμπιστης οντότητας | Προβλήματα που σχετίζονται με την ύπαρξη τρίτης έμπιστης οντότητας |
| Eschenauer, Gligor | συμμετρική | Βασίζονται στη διανομή των κρυπτογραφικών κλειδιών πριν από την ανάπτυξη του δικτύου | Υψηλές απαιτήσεις σε μνήμη για την αποθήκευση των κλειδιών σε περίπτωση μεγάλου αριθμού κόμβων |
| Karlof (TinySec) | συμμετρική | Αυθεντικοποίηση στο επίπεδο ζεύξης των δεδομένων | Κάνει χρήση ενός καθολικού κλειδιού |

# Κεφάλαιο 5

# iSense: Μια Πλατφόρμα Υλικού και Λογισμικού για Ασύρματα Δίκτυα Αισθητήρων

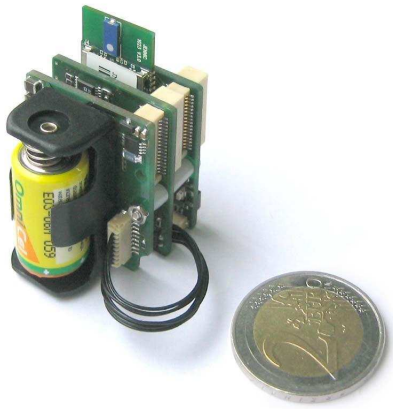
## 5.1 Εισαγωγή

Το iSense αποτελεί μια πλατφόρμα υλικού και λογισμικού για ασύρματα δίκτυα αισθητήρων.Η βασιζόμενη στα πρότυπα προσέγγιση που υποστηρίζει, επιτρέπει στο υλικό και στο λογισμικό να συμμορφώνονται πλήρως με τις απαιτήσεις της εκάστοτε εφαρμογής.Ο σχεδιασμός της πλατφόρμας που στηρίζεται στην χαμηλή κατανάλωση, επιτρέπει την εκτέλεση μιας μεγάλης αυτόνομης λειτουργίας, η οποία σε συνδυασμό με την ικανότητα για ασύρματο επαναπρογραμματισμό, καταλήγει στην εύκολη διαχείριση του δικτύου.Ο πυρήνας είναι συμβατός με το πρότυπο IEEE 802.15.4 και υποστηρίζει ZigBee ράδιο, κρυπτογράφηση στο επίπεδο του υλικού και υψηλούς ρυθμούς μετάδοσης δεδομένων.Η διεπαφή του λογισμικού είναι αρκετά ευέλικτη και παρέχει πλούσια ποικιλία προτύπων και γνωστών εργαλειών για την γρήγορη ανάπτυξη εφαργμογών.

## 5.2 Επισκόπηση του Υλικού

Η πλατφόρμα του υλικού αποτελείται από διάφορα στοιχεία (modules) τα οποία μπορούν να συνδυαστούν με διάφορους τρόπους ανάλογα με τις απαιτήσεις των εφαρμογών.Με τον τρόπο αυτό η λειτουργικότητα της κάθε εφαρμογής μπορεί να επαναπροσδιοριστεί καθώς νέα χαρακτηριστικά μπορούν να προστεθούν προσάπτοντας νέα στοιχεία υλικού.

Κατά την τρέχουσα περίοδο, τα εξής στοιχεία υλικού είναι διαθέσιμα:



Σχήμα 5.1: Μια συσκευή iSense.

- core module (υπεύθυνο για υπολογισμούς και ασύρματη επικοινωνία)
- power module
- gateway module (για σύνδεση με υπολογιστές)
- GPS module (για εύρεση της θέσης των συσκευών)
- vehicle detection module (για ανίχνευση μεγάλων μεταλλικών αντικειμένων)
- solar power system (για self powered δίκτυα)

Τα στοιχεία αυτά φαίνονται στην παρακάτω εικόνα.

Η καρδία της πλατφόρμας υλικού είναι το core module.Το στοιχείο αυτό διαχειρίζεται τον μικρο-ελεγκτή Jennic JN5139, ένα chip στο οποίο περιέχονται ο ελεγκτής και ο πομποδέκτης ασύρματης επικοινωνίας.Ο ελεγκτής υποστηρίζει 32-bit RISC υπολογισμούς και τρέχει στα 16 MHz.Αποτελείται από 96 Kb μνήμης τα οποία μοιράζονται μεταξύ του κώδικα του προγράμματος και των δεδομένων.Το πλεονέκτημα αυτής της ιδέας είναι ότι η κατανάλωση μνήμης μπορεί να εξισορροπείται ανάμεσα στα δεδομένα και τον κώδικα του προγράμματος σε αντίθεση με άλλους ελεγκτές που ο χρήστης περιορίζεται σε συγκεκριμένο μέγεθος μνήμης για δεδομένα και συγκεκριμένο για κώδικα.Ο πομποδέκτης ασύρματης επικοινωνίας είναι συμβατός με το πρότυπο ασύρματης επικοινωνίας IEEE 802.15.4.Υποστηρίζει ρυθμούς μετάδοσης δεδομένων 250 KBits/s καθώς και κρυπτογράφηση στο επίπεδο του υλικού με το βάση πρότυπο AES : Advanced Encryption Standard.Το εύρος επικοινωνίας φτάνει μέχρι τα 500m,


Σχήμα 5.2: Τα στοιχεία υλικού της πλατφόρμας iSense.

δεδομένων της ευαισθησίας λήψης περίπου -97 dBm και της ισχύς μετάδοσης ανάμεσα σε -60 dBm και +3 dBm.Εκτός από τη βασική έκδοση η οποία είναι εξοπλισμένη με SMA σύνδεσμο κεραίας, υπάρχουν συσκευές με εσωτερική κεραία για πιο συμπαγή συστήματα και συσκευές με επιπρόσθετο ενισχυτή σήματος για εύρη επικοινωνίας μέχρι 2 Km.

Ένα σύνηθες δίλημμα κατά το σχεδιασμό είναι η χρήση ή όχι ενός ρυθμιστή τάσης.Η χρήση ενός ρυθμιστή τάσης έχει το πλεονέκτημα της σωστής λειτουργίας μιας συσκευής με τάση χαμηλότερη από την απαιτούμενη.Το μειονέκτημα είναι ότι ο ρυθμιστής σπαταλάει ενέργεια και ρεύμα.Ειδικότερα στην περίπτωση που η τάση είναι μεγαλύτερη από την απαιτούμενη, ο ρυθμιστής σπαταλάει ενέργεια χωρίς η χρήση του να είναι απαραίτητη.Για την αντιμετώπιση αυτού του προβλήματος, δίνεται η δυνατότητα ενεργοποίησης του ρυθμιστή τάσης μέσω του λογισμικού.Με τον τρόπο αυτό ο ρυθμιστής τάσης αποφεύγεται όταν δεν χρειάζεται και μπορεί να ενεργοποιείται όταν η τάση είναι χαμηλή.

Για την υποστήριξη συγχρονισμού, το στοιχείο αυτό είναι εξοπλισμένο με

ένα ρολόι μεγάλης ακρίβειας.Ένας σύνδεσμος 34-pin βρίσκεται και στις δύο πλευρές του core module έτσι ώστε άλλα στοιχεία του υλικού να μπορούν να συνδεθούν με αυτό.Το core module μπορεί να παρέχει μέχρι 500 mA στα στοιχεία υλικού που συνδέονται με αυτό.

Ο ελεγκτής μπορεί να προγραμματιστεί με διάφορους τρόπους.Ο βασικός είναι ο OTAP : Over The Air Programming, αλλά επίσης το πρόγραμμα μπορεί να μεταφερθεί μέσω του gateway module(που παρουσιάζεται παρακάτω) ή μέσω ενός ειδικού προγραμματιστικού adapter που συνδέεται με το core module.

Το core module σε πλήρη λειτουργία καταναλώνει περίπου 9mA και το ράδιο του περίπου 29mA.Στην περίπτωση που μια συσκευή βρίσκεται σε κατάσταση αδράνειας η κατανάλωση μπορεί να πέσει μέχρι 10μA.Το στοιχείο αυτό μπορεί να δέχεται τροφοδοσία από την πρίζα, από κλασικές μπαταρίες, από τα power modules και από τη USB διεπαφή του gateway module.

Σχήμα 5.3: iSense Core Module.

Διάφορα power modules είναι διαθέσιμα στην πλατφόρμα iSense.Το lithium-ion module είναι ένας συνδυασμός μιας επαναφορτιζόμενης μπαταρίας μεγάλης χωρητικότητας, με ένα ελεγκτή κατανάλωσης και ένα ψηφιακό όργανο παρακολούθησης της τάσης.Το module αυτό επιτρέπει την φόρτιση της μπαταρίας των συσκευών με την σύνδεση του συστήματος στην πρίζα με ειδικό επαφέα καθώς και την παροχή της πληροφορίας της τρέχουσας ενέργειας που έχει παραμείνει στην μπαταρία.Το coin cell module είναι σχεδιασμένο για συγκεκριμένα συμπαγή συστήματα.Αποτελείται από μία μπαταρία CR2477 και ένα όργανο παρακολούθησης της ακριβής κατανάλωσης της μπαταρίας.Τέλος, το module iSense 1/2AA Battery αποτελείται από μία βάση μπαταρίας 1/2AA και μια μικρή οθόνη για την παροχή πληροφοριών όπως ενέργεια που έχει καταναλωθεί και ενέργεια που έχει απομείνει.

Το gateway module επιτρέπει τη σύνδεση σε υπολογιστές καθώς και σε άλλα δίκτυα.Εξουσιοδοτεί την ανταλλαγή δεδομένων καθώς και τον προγραμματισμό των συνδεδεμένων core modules.Εκτός από διάφορα LEDS, κουμπιά

Σχήμα 5.4: Το power module 1/2AA Battery της πλατφόρμας iSense.

Σχήμα 5.5: Τα power modules της πλατφόρμας iSense.

και ένα ποτενσιόμετρο παρέχει διεπαφές USB και RS232.

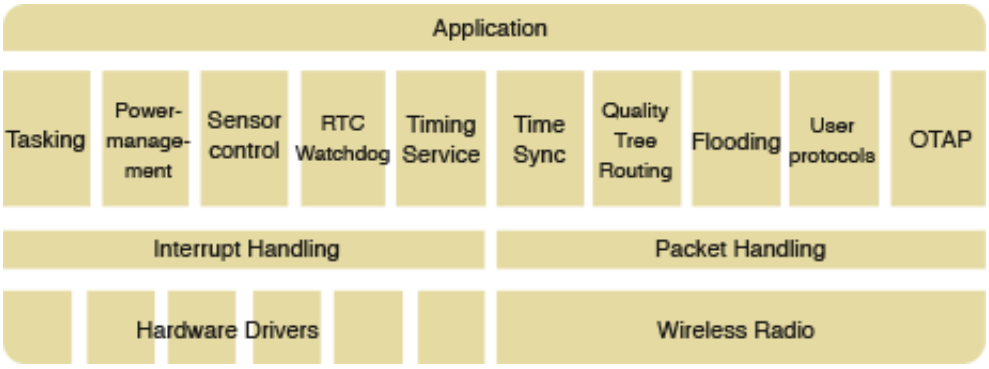
## 5.3 Επισκόπηση του Λογισμικού

Το λογισμικό που εκτελείται σε κάθε κόμβο του δικτύου απαιτεί την ίδια ευελιξία σχεδιασμού με την αντίστοιχη του υλικού της πλατφόρμας iSense.Ένας από τους βασικούς στόχους σχεδιασμού λογισμικού είναι η χρήση προγραμματιστικών μεθόδων που είναι ευρέως διαδεδομένες και αντιληπτές από την ερευνητική κοινότητα.Προηγμένες τεχνικές, όπως ο αντικειμενοστραφής C++ προγραμματισμός και η δυναμική ανάθεση μνήμης, επιτρέπουν τη γρήγορη και σωστή ανάπτυξη εφαρμογών.Τέτοιες τεχνικές συνήθως δεν είναι διαθέσιμες σε δίκτυα

Σχήμα 5.6: iSense Gateway Module με USB καλώδιο.

αισθητήρων.Η πλατφόρμα iSense υποστηρίζει μια παρόμοια με STL υλοποίηση η οποία παρέχει μεθόδους για χρήση lists,sets και maps.Έτσι η ανάπτυξη εφαρμογών στην πλατφόρμα iSense βασίζεται πλήρως σε γνωστές τεχνολογίες.

Όπως και το υλικό, αντίστοιχα και το λογισμικό της πλατφόρμας iSense οργανώνεται σε ένα σύνολο από στοιχεία (modules) καθένα από τα οποία προσφέρει μια εξειδικευμένη υπηρεσία στην εφαρμογή.Όταν ένας χρήστης αναπτύσει μια εφαρμογή, συναθροίζει τα στοιχεία σε ένα λειτουργικό σύστημα που παρέχει τις επιθυμητές λειτουργίες.Η επιλογή αυτών των λειτουργιών όπως η υποστήριξη συγκεκριμένων modules, υποστήριξη διαφόρων πρωτοκόλλων και αριθμών κινητής υποδιαστολής μπορεί να γίνει από τον κάθε χρήστη κατά το web-configuration.Η παρακάτω εικόνα παρουσιάζει την αρχιτεκτονική του λογισμικού της πλατφόρμας iSense η οποία αποτελείται από τέσερα διακριτά επίπεδα: το αφαιρετικό επίπεδο του υλικού, το επίπεδο του λειτουργικού συστήματος, το επίπεδο δικτυακής υποστήριξης και το επίπεδο των εφαρμογών των χρηστών.




Σχήμα 5.7: Η δομή του λογισμικού της πλατφόρμας iSense.

Το αφαιρετικό επίπεδο του υλικού (HAL: Hardware Abstraction Layer) συμπεριλαμβάνει τις λειτουργίες του και αποκρύπτει από τους προγραμματιστές τις περίπλοκες λεπτομέρειες του, παρέχοντας τους μια διεπαφή για τα παραπάνω επίπεδα.Το επίπεδο αυτό παρέχει αφαιρέσεις για αλληλεπίδραση με μετατροπείς A/D και D/A, με I/O διεπαφές(π.χ. UARTs,SPI ), με timers καθώς και με τον πομποδέκτη ασύρματης επικοινωνίας.Ένα τυπικό σενάριο χρήσης του HAL περιλαμβάνει την ενσωμάτωση των στοιχείων υλικού, τα οποία είναι συνήθως συνδεδεμένα σε ένα από τα I/O pins, όπως και την χρήση τους μέσω των λειτουργιών αυτού του επιπέδου.Με την αρχιτεκτονική αυτή, όλα τα modules πάνω από το HAL είναι ανεξάρτητα της συγκεκριμένης πλατφόρμας υλικού.Ο κώδικας εφαρμογής που αναπτύσεται σε αυτό το πλαίσιο είναι έτοιμος να τρέξει σε οποιαδήποτε πλατφόρμα παρέχει υλοποίηση της iSense διεπαφής.Τη δεδομένη χρονική στιγμή, ένας προγραμματιστής μπορεί να τρέξει την εφαρμογή του στην πλατφόρμα υλικού iSense ή στο εργαλείο εξομοίωσης Shawn.Με τον τρόπο αυτό μπορεί να εξετάστεί η απόδοση και η λειτουργικότητα της εφαρμογής σε επίπεδο εξομοίωσης πριν την ανάπτυξη της σε πραγματικές συσκευές.

Πάνω από το αφαιρετικό επίπεδο του υλικού βρίσκεται το λειτουργικό σύστημα το οποίο διευκολύνει την ανάπτυξη εφαρμογών μέσω ενός προσανατολισμένου στα γεγονότα μοντέλου.Μια εφαρμογή ειδοποιείται καταλλήλως όταν λαμβάνει χώρα ένα γεγονός για το οποίο ενδιαφέρεται.Ένα γεγονός μπορεί να συμβεί μέσω της εφαρμογής (π.χ. παρέρχεται ένας timer) ή μέσω του υλικού (π.χ. λήψη δεδομένων στα I/O συστήματα, αλλαγή του σήματος στην είσοδο ενός A/D μετατροπέα).Για τα γεγονότα που λαμβάνουν χώρα μέσω της εφαρμογής το λειτουργικό σύστημα παρέχει δύο επιλογές: σε λειτουργίες που απαιτείται υψηλή ακρίβεια χρόνου, μια χρονό-υπηρεσία επιτρέπει στις ειδοποιήσεις να γίνουν αμέσως ενώ σε λειτουργίες που δεν απαιτείται ακρίβεια χρόνου οι ειδοποιήσεις μπορούν να γίνουν και αργότερα.Τέλος, το λειτουργικό σύστημα είναι υπεύθυνο για την διατήρηση των ενεργειακών πόρων των κόμβων του δικτύου, όταν αυτό είναι δυνατό.Εαν είναι επιθυμητό από τον χρήστη, η υποδομή για την διαχείριση της ενέργειας μπορεί να θέσει τις συσκευές σε λειτουργία χαμηλής κατανάλωσης.

Εκτός από την λειτουργικότητα του κάθε κόμβου, ένα βασικό συστατικό των δικτύων αισθητήρων είναι η ασύρματη επικοινωνία.Το επίπεδο HAL παρέχει κατάλληλες αφαιρέσεις της ασύρματης διεπαφής πάνω από την οποία, το επίπεδο δικτυακής υποστήριξης παρέχει δυναμικές υπηρεσίες όπως δρομολόγηση(routing), συγχρονισμό (time synchronization) και over the air προγραμματιστικά modules.Η πλατφόρμα iSense παρέχει δύο υλοποιήσεις δρομολόγησης που καλύπτουν το μεγαλύτερο κομμάτι του χώρου σχεδίασης εφαρμογών για δίκτυα αισθητήρων.Η πρώτη είναι η υλοποίηση ελεγχόμενης πλημμύρας (flooding), η οποία αποτελεί μια σταθερή και ανθεκτική στα λάθη μέθοδο για τη μεταβίβαση δεδομένων, σε ένα σύνολο κόμβων, οι οποίοι βρίσκονται σε γει-

τονιά n βημάτων από τον κόμβο-αποστολέα.Η δεύτερη αποτελεί μια υλοποίηση ενός δέντρου δρομολόγησης το οποίο επιτρέπει τη μεταφορά των δεδομένων του δικτύου σε έναν ή περισσότερους σταθμούς βάσης.Στο σχήμα αυτό το βασικό μέτρο για την επιλογή των συνδέσμων είναι ο ρυθμός απώλειας πακέτων έτσι ώστε να διατηρούνται μονοπάτια με υψηλούς ρυθμούς παράδοσης μηνυμάτων και να αυξάνεται η αξιοπιστία του δικτύου.

Στις εφαρμογές των δικτύων αισθητήρων ο συγχρονισμός των ρολογιών των κόμβων είναι ζωτικής σημασίας έτσι ώστε λειτουργίες όπως συνάθροιση δεδομένων (data aggregation) να εκτελούνται ομαλά.Το χαρακτηριστικό αυτό ενσωματώνεται σαν module στο iSense και παρέχεται στους προγραμματιστές που χρησιμοποιούν αυτή τη λειτουργία, ακριβής συγχρονισμός των ρολογιών των κόμβων με απόκλιση λιγότερη από 1ms ανά δέκα βήματα.Ακόμα ένα πολύ σημαντικό module παρέχει τη δυνατότητα του ασύρματου επαναπρογραμματισμού ενός ήδη ανεπτυγμένου δικτύου.Αυτή η διαδικασία που ονομάζεται OTAP (Over the Air Programming) διασφαλίζει την ευέλικτη ανάπτυξη και λειτουργία των δικτύων αισθητήρων καθώς περιττεύουν οι ενσύρματες συνδέσεις και δεν απαιτείται μαζικός προγραμματισμός των συσκευών.

Εκτός από τα χαρακτηριστικά που παρέχει η πλατφόρμα iSense ένα διαδεδομένο και ευρέως αποδεκτό περιβάλλον ανάπτυξης είναι απαραίτητο για την δημιουργία επιτυχών εφαρμογών.Το λογισμικό του iSense χρησιμοποιεί διάσημα εργαλεία ανάπτυξης όπως ο GCC: GNU Compiler Collection και το εργαλείο ανάπτυξης Eclipse.Επιπλέον, το iSense παρέχει το iShell, ένα κατάλληλο μέσο για την αλληλεπίδραση με το δίκτυο αισθητήρων.Το εργαλείο αυτό συνδυάζει τη λειτουργικότητα ενός σειριακού τερματικού με προγραμματισμό OTAP (Over the Air Programming) των αισθητήρων.Επιπροσθέτως, παρέχει ευέλικτα plug-in συστήματα για την ενσωμάτωση λειτουργιών ορισμένων από τον χρήστη όπως ανάλυση δεδομένων και παρακολούθηση της ασύρματης επικοινωνίας στο δίκτυο.Τέλος, το iSense και το iShell παρέχουν προαιρετικά πολυπλεγμένες υπηρεσίες έτσι ώστε οι εφαρμογές να μπορούν να χρησιμοποιούν ανεξάρτητες ροές δεδομένων.

## 5.4 Πλεονεκτήματα της Πλατφόρμας iSense

Με βάση όσα αναφέραμε παραπάνω συνοψίζουμε τα πλεονεκτήματα της πλατφόρμας iSense σε σχέση με άλλες πλατφόρμες ασυρμάτων δικτύων αισθητήρων.

- Η πλατφόρμα iSense υποστηρίζει μια πρότυπη προγραμματιστική γλώσσα (C++ like) η οποία είναι προσανατολισμένη στα αντικείμενα.

- Η πλατφόρμα iSense υποστηρίζει δυναμική ανάθεση μνήμης.

- Το λειτουργικό σύστημα της πλατφόρμας iSense παρέχει πλούσιες λειτουργικότητες.
- Υπάρχουν αρκετά έτοιμα υλοποιημένα πρωτόκολλα, όπως πρωτόκολλα δρομολόγησης, μεταφοράς, συγχρονισμού και εντοπισμού.
- Υπάρχει διαθέσιμο λογισμικό για τα διάφορα modules του υλικού.
- Ο κώδικας των εφαρμογών της πλατφόρμας iSense μπορεί να εκτελεστεί στον εξομοιωτή δικτύων αισθητήρων Shawn.Αυτό βοηθάει σημαντικά στην εξέταση της λειτουργικότητας των εφαρμογών και την αποσφαλμάτωση πριν την ανάπτυξη τους σε πραγματικές συσκευές.

# Κεφάλαιο 6

# Ανάπτυξη Πρωτοκόλλου Διαχείρισης Δημοσίου Κλειδιού με Ελλειπτικές Καμπύλες στην Πλατφόρμα iSense

## 6.1 Εισαγωγή

Σκοπός αυτής της διπλωματικής εργασίας είναι η ανάπτυξη ενός κρυπτογραφικού πρωτοκόλλου με χρήση της κρυπτογράφησης ελλειπτικών καμπυλών (ECC: Elliptic Curve Cryptography) σε ασύρματα δίκτυα αισθητήρων.Η ανάπτυξη του πρωτοκόλλου έγινε στην πλατφόρμα iSense η οποία προσφέρει ένα εναρμονισμένο περιβάλλον υλικού και λογισμικού για ανάπτυξη εφαρμογών σε δίκτυα αισθητήρων.Για την υλοποίηση της κρυπτογράφησης ελλειπτικών καμπυλών αρχικά έγιναν προσπάθειες ενσωμάτωσης της βιβλιοθήκης ECC-LIB στην πλατφόρμα iSense οι οποίες δεν είχαν αποτέλεσμα για λόγους που θα αναφέρουμε στη συνέχεια.Λόγω του γεγονότος αυτού, βασιστήκαμε στην εργασία του David J.Malan η οποία αποτελεί μια από τις πρώτες υλοποιήσεις στον τομέα αυτό και την επεκτείναμε αναπτύσοντας μηχανισμούς κρυπτογράφησης/αποκρυπτογράφησης δεδομένων.Συγκεκριμένα, στο πρωτόκολλο που υλοποιήσαμε οι συσκευές του δικτύου αρχικά εκτελούν τη διαδικασία συμφωνίας κλειδιού Diffie-Hellman με ελλειπτικές καμπύλες, στη συνέχεια ανταλλάσουν μπλοκ δεδομένων κρυπτογραφημένα με το κλειδί αυτό και τέλος τα αποκρυπτογραφούν.Η λειτουργικότητα του πρωτοκόλλου κατά τη διάρκεια ανάπτυξης του, δοκιμάστηκε με τη χρήση του εργαλείου Shawn που αποτελεί έναν εξομοιωτή ασυρμάτων δικτύων

αισθητήρων.Η απόδοση του εξετάστηκε ως προς το παράγοντα της χρονικής καθυστέρησης σε πραγματικές συσκευές της σειράς iSense και συγκρίθηκε με αυτή προηγούμενων εργασιών.

## 6.2 Η βιβλιοθήκη ECC-LIB

Η ECC-LIB αποτελεί μια μεταφερόμενη βιβλιοθήκη λογισμικού που αποτελείται από πρότυπα στοιχεία και στοχεύει στην υλοποίηση κρυπτογραφίας ελλειπτικών καμπυλών.Η βιβλιοθήκη αυτή διευκολύνει την ανάπτυξη κρυπτογραφικών πρωτοκόλλων βασιζόμενα στις ελλειπτικές καμπύλες καθώς παρέχει τις βασικές αλγεβρικές πράξεις με ελλειπτικές καμπύλες και τις πιο γνωστές μεθόδους για γέννηση ασφαλών καμπυλών.Οι δημιουργοί της, Κωνσταντίνου Ε., Σταματίου Γ. και Ζαρολιάγκης Χ., την υλοποίησαν σε ANSI C και βασίστηκαν στην GMP (GNU Multiple Precission Library) για αριθμητική αυθαίρετης ακρίβειας πάνω σε ακέραιους και αριθμούς κινητής υποδιαστολής.Ο πηγαίος κώδικας της βιβλιοθήκης παρέχεται και διανέμεται υπό την άδεια λογισμικού GPL (General Public License).

### 6.2.1 GNUMP (GNU Multiple Precission): Αριθμητική Χωρίς Όρια

#### Τι είναι η GNUMP

Η GMP είναι μια ελεύθερη βιβλιοθήκη η οποία προσφέρει αριθμητική αυθαίρετης ακρίβειας πάνω σε προσημασμένους ακέραιους, λογικούς αριθμούς και αριθμούς κινητής υποδιαστολής.Τα μόνα πρακτικά όρια που υπάρχουν για την ακρίβεια με την οποία πραγματοποιούνται οι πράξεις, είναι αυτά που τίθενται από την διαθέσιμη μνήμη της μηχανής πάνω στην οποία εκτελείται η GMP.Η GNUMP παρέχει ένα πλούσιο σύνολο συναρτήσεων οι οποίες έχουν κατάλληλες διεπαφές.Η βιβλιοθήκη αυτή βρίσκει εφαρμογή σε διάφορα ερευνητικά πεδία όπως κρυπτογραφία, ασφάλεια στο διαδίκτυο και διάφορα αλγεβρικά συστήματα.Δύο γνωστές κρύπτο-βιβλιοθήκες που χρησιμοποιούν τη GMP είναι οι LiDIA, μια C++ βιβλιοθήκη γύρω από τη θεωρία αριθμών και η ECC-LIB που θα περιγράψουμε παρακάτω.Η GMP είναι προσεκτικά σχεδιασμένη έτσι ώστε να είναι όσο πιο γρήγορη γίνεται τόσο για μικρά όσο και για μεγάλα ορίσματα.Η ταχύτητα επιτυγχάνεται χρησιμοποιώντας λέξεις (fullwords) ως βασικό αριθμητικό τύπο και γρήγορους αλγορίθμους με ενσωματωμένο κώδικα σε γλώσσα μηχανής (assembly) για την βέλτιστη εκτέλεση εσωτερικών βρόχων σε ποικίλες αρχιτεκτονικές.Η GMP είναι γρηγορότερη από κάθε άλλη βιβλιοθήκη μεγάλων αριθμών και έχει το πλεονέκτημα ότι είναι ακόμα πιο γρήγορη (σε

σχέση με τις άλλες βιβλιοθήκες) όσο το μέγεθος των ορισμάτων αυξάνει, καθώς χρησιμοποιεί ασυμπτωτικά πιο αποδοτικούς αλγόριθμους.Η πρώτη διανομή της βιβλιοθήκης GMP έγινε το 1991 και έκτοτε αναπτύσεται και βελτιώνεται συνεχώς παρέχοντας μια νέα έκδοση κάθε χρόνο.Οι χρήστες της έχουν το δικαίωμα ελεύθερης χρήσης όπως και το δικαίωμα βελτίωσης και διαμοιρασμού καθώς διανέμεται υπό την άδεια λογισμικού GNU LGPL.Οι βασικές πλατφόρμες πάνω στις οποίες μπορεί να εκτελεστεί η GNUMP είναι τα συστήματα τύπου UNIX όπως GNU/Linux, Solaris, HP-UX, Mac OS, BSD, AIX κτλ.Ακόμα λειτουργεί και σε 32-bit Windows πλατφόρμες.

### Κατηγορίες Συναρτήσεων που Προσφέρει η GNUMP

Υπάρχουν ποικίλες κατηγορίες συναρτήσεων που υποστηρίζει η βιβλιοθήκη GMP.Αυτές αναφέρονται παρακάτω:

- Αριθμητικές Συναρτήσεις Προσημασμένων Ακεραίων Αριθμών Υψηλού Επιπέδου (mpz): Υπάρχουν πάνω από 140 λογικές (xor,and) και αριθμητικές (πρόσθεση, διαίρεση, εύρεση μέγιστου κοινού διαιρέτη) συναρτήσεις σε αυτήν την κατηγορία.

- Αριθμητικές Συναρτήσεις Λογικών Αριθμών (mpq): Η κατηγορία αυτή αποτελείται από 35 συναρτήσεις για την υποστήριξη αριθμητικής πάνω σε λογικούς αριθμούς.

- Αριθμητικές Συναρτήσεις Αριθμών Κινητής Υποδιαστολής (mpf): Η κατηγορία αυτή παρέχει 65 συναρτήσεις για αριθμητική πάνω σε αριθμούς κινητής υποδιαστολής και είναι κατάλληλη για εφαρμογές που ο τύπος double της γλώσσας C δεν παρέχει ικανοποιητική ακρίβεια.

- C++ Διεπαφή : Η GMP παρέχει μια διεπαφή βασισμένη σε κλάσεις για την υποστήριξη όλων των παραπάνω συναρτήσεων.

- Αριθμητικές Συναρτήσεις Θετικών Ακεραίων Χαμηλού Επιπέδου (mpn): Οι συναρτήσεις της κατηγορίας αυτής δεν εκτελούν διαχείριση μνήμης οπότε ο χρήστης πρέπει να διαβεβαιώνεται ότι το μέγεθος της διαθέσιμης μνήμης είναι επαρκές.Οι συναρτήσεις mpn δέχονται ορίσματα στη μορφή ζευγαριών.Το κάθε ζευγάρι αποτελείται από ένα δείκτη στη λιγότερο σημαντική λέξη του ορίσματος και έναν αριθμό που δηλώνει από πόσες λέξεις αποτελείται το όρισμα.Οι συναρτήσεις των παραπάνω κατηγορίων καλούν τις συναρτήσεις mpn σχεδόν για κάθε υπολογισμό που εκτελούν.

- Συναρτήσεις συμβατές με αυτές της αριθμητικής αυθαίρετης ακρίβειας που έφτιαξε ομάδα του πανεπιστημίου Berkeley.

- Εξωτερική υποστήριξη αριθμητικών συναρτήσεων στογγυλοποίησης (mpfr) των αριθμών κινητής υποδιαστολής.

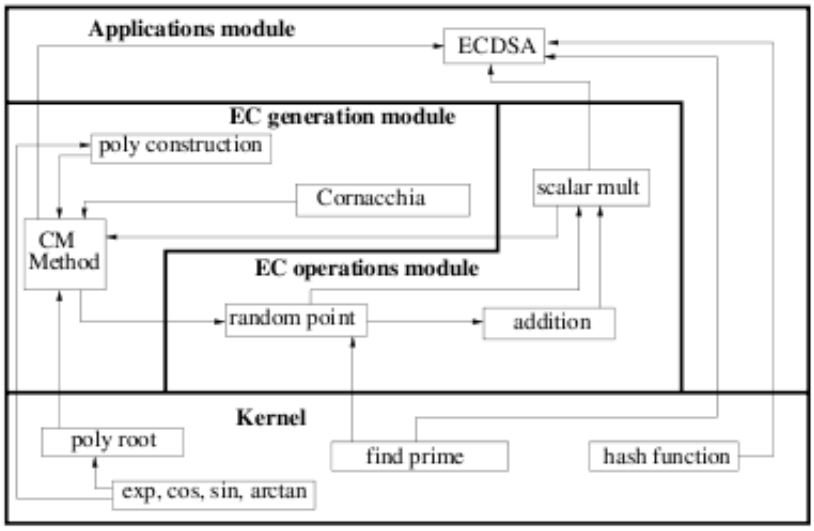
### 6.2.2 Θέματα Σχεδιασμού της Βιβλιοθήκης ECC-LIB

Όπως αναφέρθηκε παραπάνω, βασικός στόχος της βιβλιοθήκης ECC-LIB είναι η μεταφερσιμότητα και η ευκολία χρήσης.Κατά την ανάπτυξή της, οι δημιουργοί της Χ.Ζαρολιάγκης, Γ.Σταματίου και Ε.Κωνσταντίνου, πήραν αποφάσεις σχετικά με το πεδίο των ελλειπτικών καμπυλών, με το μέγεθος του, με τις μεθόδους γέννησης ελλειπτικών καμπυλών και με τις βιβλιοθήκες που θα χρησιμοποιούσαν για αριθμητική πάνω σε μεγάλους αριθμούς.

Για την αντιμετώπιση της μεταφερσιμότητας, η βιβλιοθήκη γράφτηκε σε ANSI C χρησιμοποιώντας τη GMP (GNU Multiple Precission Library) για αριθμητική μεγάλης ακρίβειας.Όσον αφορά στην υλοποίηση των ελλειπτικών καμπυλών επιλέχτηκε το πεδίο πρώτων αριθμών $F_p$ λόγω της απλότητας του στην αναπαράσταση και στις αλγεβρικές πράξεις.Για την αναπαράσταση των αριθμών στο πεδίο πρώτων αριθμών $F_p$ χρησιμοποιήθηκαν οι μεγάλοι αριθμοί που προσφέρει η GMP.Η βιβλιοθήκη αυτή αναπαριστά τους ακέραιους και τους αριθμούς κινητής υποδιαστολής χρησιμοποιώντας μια μονάδα που ονομάζεται limb και αποτελείται από 32-bits.Εκτός από τις συναρτήσεις που προσφέρει η GMP υλοποιήθηκαν βασικές αλγεβρικές πράξεις των μιγαδικών αριθμών (πρόσθεση, πολλαπλασιασμός, δύναμη) καθώς και κάποιες συναρτήσεις για τους αριθμούς κινητής υποδιαστολής όπως $\cos(x)$, $\sin(x)$, $\ln(x)$, $\sqrt{x}$ και $\arctan(x)$.Οι συναρτήσεις αυτές είναι απαραίτητες για μεθόδους όπως η Complex Multiplication η οποία γεννά μια καμπύλη κατάλληλης τάξης και υπολογίζει τις μεταβλητές $a$ και $b$.

Η αρχιτεκτονική της βιβλιοθήκης ECC-LIB αποτελείται από τέσσερα βασικά επίπεδα: τον πυρήνα (kernel), το επίπεδο πράξεων ελλειπτικών καμπυλών (EC operations module), το επίπεδο γέννησης ελλειπτικών καμπυλών (EC generation module) και το επίπεδο εφαρμογών (applications module).Τα βασικά στοιχεία της αρχιτεκτονικής της βιβλιοθήκης φαίνονται στην παρακάτω εικόνα.

Ο πυρήνας αποτελείται από διάφορα στοιχεία που υλοποιούν βασικές αλγεβρικές και τριγωνομετικές πράξεις πάνω σε ακέραιους και αριθμούς κινητής υποδιαστολής.Επιπλέον, παρέχει και κάποιες πιο εξειδικευμένες συναρτήσεις που δημιουργήθηκαν από την αρχή, όπως χειρισμός ακεραίων συντελεστών πολυωνύμων και εύρεση των ριζών ενός πολυωνύμου modulo ενός πρώτου αριθμού.Όλα τα στοιχεία που αποτελούν τον πυρήνα είναι ανεξάρτητα από τα στοιχεία των ανωτέρων επιπέδων που τα χρησιμοποιούνε, έτσι ώστε να υπάρχει η δυνατότητα ανεξάρτητης βελτίωσης τους για καλύτερη απόδοση της βιβλιοθήκης.




Σχήμα 6.1: Η αρχιτεκτονική της βιβλιοθήκης ECC-LIB.

Το EC operations module αποτελείται από στοιχεία που υλοποιούν τις βασικές αλγεβρικές πράξεις πάνω σε ελλειπτικές καμπύλες.Ένα στοιχείο ορίζει τον τύπο δεδομένων της ελλειπτικής καμπύλης (ένας πίνακας που αποτελείται απο δύο αριθμούς άπειρης ακρίβειας οι οποίοι αναπαριστούν τις μεταβλητές $a$ και $b$ της καμπύλης) και μία δομή αναπαριστά ένα σημείο πάνω σε μια καμπύλη ως ένα ζευγάρι ακεραίων άπειρης ακρίβειας.Τέλος, υπάρχουν στοιχεία τα οποία δημιουργούν τυχαία σημεία πάνω σε μια καμπύλη, εκτελούν την πρόσθεση δύο σημείων πάνω στην ελλειπτική καμπύλη, τον βαθμωτό πολλαπλασιασμό ενός σημείου με έναν ακέραιο καθώς και τη γέννηση ενός σημείου γεννήτορα πάνω στην καμπύλη.

Το EC generation module είναι το πιο σημαντικό στοιχείο της βιβλιοθήκης ECC-LIB.Αποτελείται από διάφορα στοιχεία που υλοποιούν μεθόδους όπως τον αλγόριθμο του Cornacchia για την επίλυση διοφαντικών εξισώσεων και την Complex Multiplication για τη γέννηση ασφαλών καμπυλών με χρήση πολυωνύμων Weber και Hilbert.

Τέλος, το applications module περιέχει διάφορα κρυπτογραφικά πρωτόκολλα υψηλού επιπέδου και μεθόδους που βασίζονται στις ελλειπτικές καμπύλες.Μερικά από αυτά είναι το πρωτόκολλο ανταλλαγής κλειδιού Diffie-Hellman, οι μέθοδοι γέννησης ιδιωτικού και δημόσιου κλειδιού, κρυπτογράφηση/αποκρυπτογράφηση δεδομένων και ο αλγόριθμος ψηφιακών υπογραφών ECDSA.Έτσι, υπάρχει η δυνατότητα δημιουργίας πλουσιότερων κρυπτογραφικών πρωτοκόλλων βασιζόμενος στα στοιχεία που του παρέχει η βιβλιοθήκη ECC-LIB.

### 6.2.3 Πειραματικά Αποτελέσματα

Για την εξέταση της απόδοσης της βιβλιοθήκης ECC-LIB έγινε πειραματική μελέτη πάνω στα βασικά της στοιχεία.Τα πειράματα εκτελέστηκαν σε υπολογιστή Pentium III (933 Mhz) με κύρια μνήμη 256Mb.Όπως προαναφέρθηκε χρησιμοποιήθηκε η βιβλιοθήκη GNUMP για αριθμητική αυθαίρετης ακρίβειας και ο μεταφραστής της ANSI-C gcc-2.95.2.Παρακάτω παρουσιάζονται οι επεξεργαστικοί χρόνοι του βαθμωτού πολλαπλασιασμού και των βασικών κρυπτογραφικών πρωτοκόλλων που υλοποιήθηκαν με χρήση της βιβλιοθήκης.Η μεταβλητή $p$ αποτελεί έναν πρώτο αριθμό, η $|p|$ αποτελεί το μέγεθος του και η $p_{|p|}$ αναπαριστά έναν πρώτο αριθμό μεγέθους $|p|$.Τα πειράματα έγιναν σε τρία διαφορετικά πεδία με διαφορετικά μεγέθη : $F_{p_{175}}$, $F_{p_{192}}$ και $F_{p_{224}}$.

Πίνακας 6.1: Επεξεργαστικός χρόνος σε msec των διαφόρων στοιχείων της βιβλιοθήκης ECC-LIB.

| $\|p\|$ | 175bits | 192bits | 224bits |
|---|---|---|---|
| Scalar Multiplication | 13.6 | 15.7 | 19.5 |
| Key Generation | 19.6 | 23.9 | 30.8 |
| ECDH protocol | 27.2 | 31.4 | 39 |
| ECES encryption | 28.8 | 36.5 | 46 |
| ECES decryption | 13.5 | 16.3 | 19.1 |
| ECDSA Signature | 19.1 | 22.7 | 30.6 |
| ECDSA Verify | 24.5 | 28.3 | 36.8 |

Θα πρέπει να επισημάνουμε ότι δεν έγινε κάποια προσπάθεια για βελτιστοποίηση του κώδικα γράφοντας π.χ κομμάτια της βιβλιοθήκης σε γλώσσα μηχανής.Όπως παρατηρείται και στον παραπάνω πίνακα οι χρόνοι εκτέλεσης αυξάνουν όσο αυξάνει το μέγεθος του πεδίου.

Στα πλαίσια αυτής της εργασίας έγιναν προσπάθειες ενσωμάτωσης της βιβλιοθήκης ECC-LIB στην πλατφόρμα iSense.Οι προσπάθειες αυτές απέτυχαν για δύο βασικούς λόγους.Ο πρώτος είναι η περιορισμένη διαθέσιμη μνήμη που παρέχουν οι συσκευές αισθητήρων.Συγκεκριμένα, οι συσκευές iSense παρέχουν 96Kb μνήμης που μοιράζονται μεταξύ των δεδομένων και του κώδικα του προγράμματος το οποίο εκτελούν.Το γεγονός ότι η βιβλιοθήκη ECC-LIB βασίζεται στη βιβλιοθήκη GNUMP για την υποστήριξη αριθμητικής άπειρης ακρίβειας δεν επιτρέπει την ενσωμάτωση της στη συσκευή iSense κι αυτό διότι οι απαιτήσεις σε μνήμη είναι τεράστιες (σε σχέση με τη μνήμη που παρέχει η συσκευή).Ο δεύτερος, αφορά στο λειτουργικό συστημα που παρέχει η πλατφόρμα iSense.Συγκεκριμένα, το λειτουργικό σύστημα της πλατφόρμας δεν υποστηρίζει κάποια standard χαρακτηριστικά της γλώσσας C τα οποία χρησιμοποιεί η GMP όπως π.χ. stdout και malloc .Αντιθέτως υποστηρίζει τα αντίστοιχα δικά

του όπως π.χ. Os::debug και isense::malloc.Το γεγονός αυτό συνδυαζόμενο με το γεγονός ότι η GMP μπορεί να εκτελεστεί σε συγκεκριμένες πλατφόρμες καθιστά αδύνατη τη χρήση της σε συσκευές iSense.Μια πιθανή λύση στο πρώτο πρόβλημα θα ήταν η απομόνωση των συναρτήσεων των βιβλιοθηκών ECC-LIB και GMP που μας ενδιαφέρουν για την μείωση της απαιτούμενης μνήμης.Όμως και πάλι λόγω του δεύτερου προβλήματος σε συνδυασμό με το γεγονός της υλοποίησης κάποιων GNUMP λειτουργιών σε γλώσσα μηχανής, απαιτούνται αρκετές αλλαγές στις συναρτήσεις αυτές για την σωστή εκτέλεση τους στην πλατφόρμα iSense.Έτσι, τελικά για την υλοποίηση κρυπτογραφίας δημοσίου κλειδιού με ελλειπτικές καμπύλες βασιστήκαμε στην εργασία του David J.Malan που περιγράφεται στη συνέχεια.

## 6.3 Η Εργασία του David J.Malan

Το 2004 οι David Malan, Matt Welsh και Michael Smith παρουσίασαν την πρώτη υλοποίηση κρυπτογραφίας με ελλειπτικές καμπύλες, ορισμένες πάνω από τα δυαδικά πεδία $F_{2^p}$, σε ασύρματα δίκτυα αισθητήρων.Συγκεκριμένα, η υλοποίηση αυτή έγινε για τις συσκευές MICA 2 οι οποίες περιέχουν 8-bit επεξεργαστή που τρέχει στα 7.3828 MHz.Ο βασικός στόχος της εργασίας αυτής ήταν να καλύψει την ανάγκη ύπαρξης ενός ασφαλούς μηχανισμού για τη διανομή κλειδιών ανάμεσα στους κόμβους, παρά το γεγονός ότι η υποδομή δημόσιου κλειδιού θεωρούνταν μη-πρακτική την εποχή εκείνη.Απέδειξαν ότι η συμμετρική κρυπτογραφία είναι βιώσιμη στις συσκευές MICA 2 και υποστηρίξανε με την υλοποίηση πολλαπλασιασμού σημείων πάνω σε μια ελλειπτική καμπύλη ότι η υποδομή δημοσίου κλειδιού είναι ικανοποιητική για τη διανομή των μυστικών κλειδιών στους κόμβους.

Το TinyOS προσφέρει στις συσκευές MICA 2 έλεγχο πρόσβασης, πιστοποίηση, ακεραιότητα και εμπιστευτικότητα μέσω του TinySec.Το TinySec αποτελεί ένα μηχανισμό ασφάλειας στο επίπεδο συνδέσμου ο οποίος βασίζεται στο Skipjack.Ο Skipjack είναι ένας αλγόριθμος κρυπτογράφησης που αναπτύχθηκε από το NIST : National Institute for Standards and Technology και χρησιμοποιεί ένα κλειδί μήκους 80-bit για την κρυπτογράφηση 64-bit μπλοκ δεδομένων.Για τον κατάλληλο διαμοιρασμό των 80-bit κλειδιών του TinySec χρειάζεται ένας μηχανισμός αντίστοιχης ασφάλειας.Έχουμε αναφέρει και σε προηγούμενα κεφάλαια ότι με το πρωτόκολλο Diffie-Hellman δύο κόμβοι μπορούν να συμφωνήσουν σε ένα κοινό μυστικό.Σύμφωνα με το NIST για τον ασφαλή διαμοιρασμό κλειδιών 80-bit μέσω του πρωτοκόλλου Diffie-Hellman απαιτείται ένας πρώτος αριθμός τουλάχιστον 1024-bits και ένας εκθέτης μεγέθους 160-bits.Όπως είναι αντιληπτό, σε μια αρχιτεκτονική 8-bit όπως το MICA 2 οι υπολογισμοί αριθμών μήκους 160 και 1024 bit είναι αρκετά απαιτητικοί.

Οι συγγραφείς κατάφεραν να λύσουν το παραπάνω πρόβλημα με χρήση των ελλειπτικών καμπυλών.Η ασφαλής διανομή 80-bit κλειδιών μπορεί να γίνει με κλειδιά που βασίζονται στις ελλειπτικές καμπύλες μήκους μόλις 163-bit.Οι ελλειπτικές καμπύλες προσφέρουν μια διαφορετική βάση για την ανταλλαγή κοινών μυστικών ανάμεσα σε επιτιθέμενους με απόλυτη προς τα εμπρός ασφάλεια.Το πρόβλημα ECDLP: Elliptic Curve Discrete Logarithm Problem στο οποίο βασίζεται η κρυπτογραφία με ελλειπτικές καμπύλες ECC, περιλαμβάνει την επαναφορά ενός αριθμού $k$ πάνω από ένα πεπερασμένο πεδίο Galois $F$, δεδομένων του γινομένου $kG$, του σημείου $G$ και της εξίσωσης $E$ της ελλειπτικής καμπύλης σε μορφή Weierstrass η οποία φαίνεται παρακάτω:

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$$

όπου $a_i \in F$.

Οι κρυπτογράφοι δείχνουν ενδιαφέρον στις ελλειπτικές καμπύλες που είναι ορισμένες πάνω από τα πεδία $F_p$ και $F_{2^p}$ όπου το $p$ είναι ένας πρώτος αριθμός.Ειδικότερα, οι ελλειπτικές καμπύλες ορισμένες πάνω από τα δυαδικά πεδία είναι ιδιαίτερα δημοφιλείς καθώς προσφέρουν αποδοτικούς σε χώρο και χρόνο αλγόριθμους.

### 6.3.1 Κρυπτογραφία Ελλειπτικών Καμπυλών στο $F_{2^p}$

Η κρυπτογραφία ελλειπτικών καμπυλών πάνω από το πεδίο $F_{2^p}$ αρχικά απαιτεί την επιλογή μιας βάσης για την αναπαράσταση των σημείων έτσι ώστε κάθε $a \in F_{2^p}$ να μπορεί να γραφτεί στη μορφή

$$a = \sum_{i=0}^{m-1} a_i a_i$$

όπου $a_i \in \{0, 1\}$.Ορισμένο με τον τρόπο αυτό το $a$ μπορεί να αναπαρασταθεί ως ένας δυαδικό διάνυσμα, $\{a_0, a_1, ..., a_{p-1}\}$, όπου το $\{a_0, a_1, ..., a_{p-1}\}$ αποτελεί τη βάση πάνω από το πεδίο $F_2$.Οι πιο κοινές βάσεις είναι οι πολυωνυμικές και οι κανονικές.Όταν κάθε $a_i \in F_{2^p}$ αναπαρίσταται με πολυωνυμική βάση, αντιστοιχεί σε ένα δυαδικό πολυώνυμο βαθμού μικρότερου του p, όπως:

$$a = a_{p-1}x^{p-1} + a_{p-2}x^{p-2} + ... + a_0x^0$$

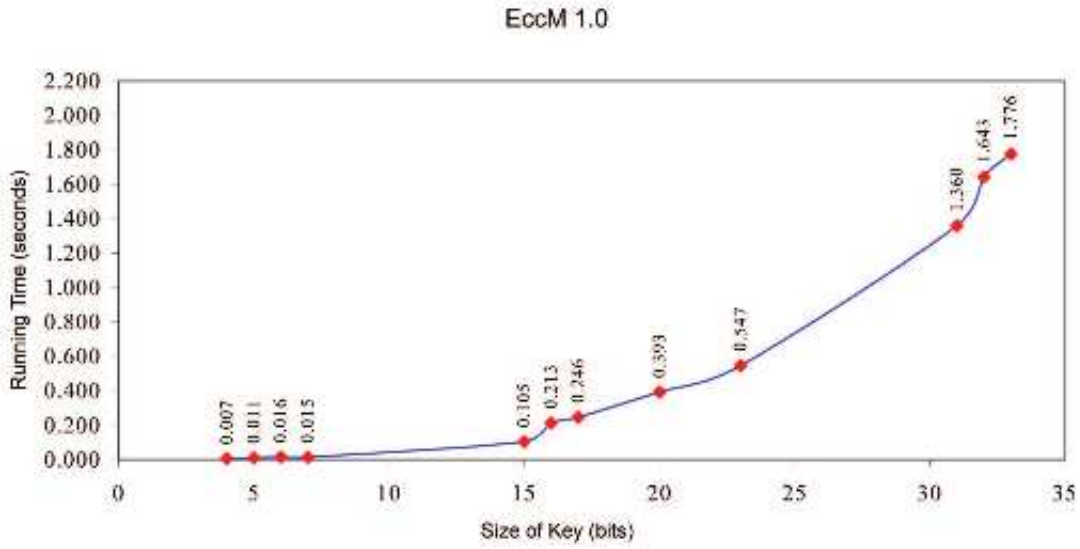
όπου και πάλι $a_i \in \{0, 1\}$.Με τον τρόπο αυτό κάθε $a \in F_{2^p}$ μπορεί να απεικονιστεί στη μνήμη της συσκευής MICA 2 σαν μια ακολουθία από bit, $a_{p-1}a_{p-2}...a_0$.Όλες οι πράξεις των στοιχείων αυτών γίνονται modulo ενός απλοποιημένου πολυωνύμου, $f$ , βαθμού $p$ πάνω από το πεδίο $F_2$, έτσι ώστε $f(x) = x^p + \sum_{i=0}^{p-1} f_i x_i$, με $f_i \in \{0, 1\}$ και $i \in \{0, 1, ..., p-1\}$.Τυπικά, εαν

υπάρχει στο $F_2$ ένα μη μειούμενο τριώνυμο $x^p + x^k + 1$, το $f(x)$ επιλέγεται έτσι ώστε να είναι αυτό με το μικρότερο $k$.Στην περίπτωση που τέτοιο πολυώνυμο δεν υπάρχει, η $f(x)$ επιλέγεται να είναι πολυώνυμο 5ου βαθμού, $x^p + x^{k3} + x^{k2} + x^{k1} + 1$, έτσι ώστε το $k1$ να είναι ελάχιστο, το $k2$ να είναι ελάχιστο ως προς το $k1$ και το $k3$ να είναι ελάχιστο ως προς τα $k1$ και $k2$.

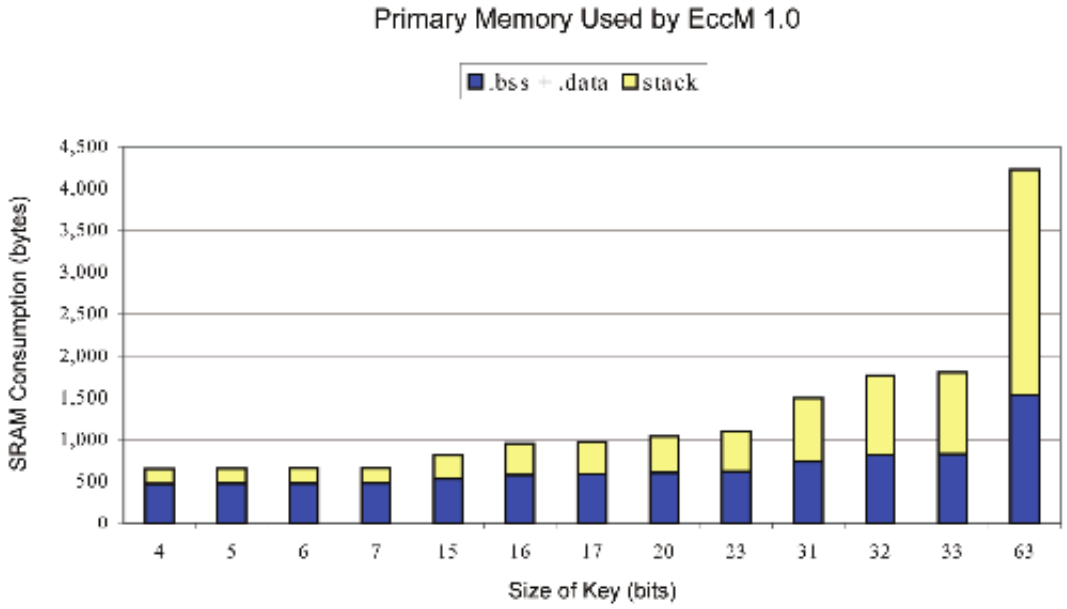
Σε μια πολυωνυμική βάση, η πρόσθεση δύο στοιχείων $a, b$ ορίζεται ως $a+b = c$, όπου $c_i = a_i + b_i(mod2)$.Ο πολλαπλασιασμός των $a, b$ ορίζεται ως $a * b = c$, όπου $c(x) = (\sum_{i=0}^{p-1} a_i x^i)(\sum_{i=0}^{p-1} b_i x^i)(modf(x))$.Οι συγγραφείς επέλεξαν μια πολυωνυμική βάση για την υλοποίηση του βαθμωτού πολλαπλασιασμού καθώς προσφέρει πιο αποδοτικές υλοποιήσεις.

### 6.3.2 Πρώτη Υλοποίηση EccM 1.0

Η πρώτη υλοποίηση που επιχείρησαν οι συγγραφείς για τις συσκευές MICA 2 ονομαζόταν EccM 1.0.Το EccM 1.0 που αποτελεί ένα module για το λειτουργικό σύστημα TinyOS, αρχικά επιλέγει μια τυχαία καμπύλη στη μορφή της εξίσωσης που αναφέραμε παραπάνω έτσι ώστε $a = 0$ και $b \in F_{2^p}$.Εν συνεχεία, υπολογίζει ένα τυχαίο σημείο $G \in F_{2^p} \times F_{2^p}$ πάνω στην καμπύλη καθώς και ένα τυχαίο αριθμό $k \in F_{2^p}$ που αποτελεί το ιδιωτικό κλειδί του κόμβου.Τέλος, υπολογίζει το γινόμενο $kG$ που είναι το δημόσιο κλειδί του κόμβου.Τα πρώτα αποτελέσματα της υλοποίησης αυτής ήταν ενθαρυντικά καθώς η δημιουργία ενός κλειδιού μήκους 33-bit απαιτούσε μόλις 1.776 δευτερόλεπτα.Παρά το γεγονός αυτό, το πρωτόκολλο απέτυχε να δημιουργήσει κλειδιά μεγαλύτερου μήκους (π.χ. 63 bit) αναγκάζοντας τον κόμβο να επανεκκινήσει λόγω υπερχείλισης της στοίβας.Τα αποτελέσματα της υλοποίησης αυτής φαίνονται στις παρακάτω γραφικές παραστάσεις.




Σχήμα 6.2: Χρόνοι δημιουργίας κλειδιού στο EccM 1.0.Για κλειδιά μήκους 63-bit το πρωτόκολλο δεν απέφερε αποτελέσματα.




Σχήμα 6.3: Κατανάλωση μνήμης στο EccM 1.0. Κλειδιά μήκους 63-bit εξαντλούν τη RAM των συσκευών MICA 2.

### 6.3.3 Δεύτερη Υλοποίηση EccM 2.0

Λόγω της αποτυχίας της πρώτης υλοποίησης στη δημιουργία κλειδιών 63-bit, οι συγγραφείς προχώρησαν στη δεύτερη που ονομάζεται EccM 2.0.Η EccM 2.0 διαλέγει έναν κόμβο Α και δημιουργεί το ιδιωτικό του κλειδί $k_A$ χρησιμοποιώντας μια πολυωνυμική βάση στο $F_{2^p}$.Στη συνέχεια αφού βρει ένα σημείο βάσης $G$ πάνω σε μια καμπύλη Koblitz, υπολογίζει το δημόσιο κλειδί του κόμβου $T_A = K_A G$.Το δημόσιο κλειδί του Α γίνεται γνωστό σε κάθε κόμβο Β με τον οποίο επιθυμείται ασφαλής επικοινωνία.Ο Α λαμβάνει με αντίστοιχο τρόπο το δημόσιο κλειδί του Β και έτσι ο καθένας τους μπορεί να υπολογίσει το κοινό μυστικό $k_A k_B G$, όπου $k_B$ είναι το ιδιωτικό κλειδί του Β.

Η υλοποίηση EccM 2.0 δεν κατάφερε μόνο τη δημιουργία δημοσίων κλειδιών 163-bit αλλά έδωσε λύσεις και σε ένα άλλο σημαντικό πρόβλημα.Στην EccM 1.0 οι ελλειπτικές καμπύλες επιλέγονταν τυχαία με κίνδυνο να είναι ευάλωτες σε επιθέσεις MOV.Αντίθετα, στην EccM 2.0 επιλέγεται μια καμπύλη που ικανοποιεί τις απαιτήσεις του NIST για ECC πάνω από το πεδίο $F_{2^p}$.Έτσι, η εξίσωση της καμπύλης είναι η:

$$y^2 + xy = x^3 + x^2 + 1$$

σαν απλοποιημένο πολυώνυμο χρησιμοποιείται το:

$$f(x) = x^{163} + x^7 + x^6 + x^3 + 1$$

η τάξη της καμπύλης ( ο αριθμός των σημείων πάνω σε αυτή) είναι:

$$0x4000000000000000000020108a2e0cc0d99f8a5ef$$

και το σημείο βάσης είναι το $G = (G_x, G_y)$ με :

$$G_x = 0x2fe13c0537bbc11acaa07d793de4e6d5e5c94eee8$$

και

$$G_y = 0x289070fb05d38ff58321f2e800536d538ccdaa3d9$$

Αυτό που κατάφεραν οι συγγραφείς με την υλοποίηση EccM -2.0 ήταν η δημιουργία μεγαλύτερων και πιο ασφαλών κλειδιών σε σχέση με αυτά της EccM-1.0 καθώς και η πολύ μικρότερη κατανάλωση μνήμης.Ο βαθμωτός πολλαπλασιασμός σημείου της ελλειπτικής καμπύλης χρονομετρήθηκε γύρω στα 34 δευτερόλεπτα με απόκλιση 0.9 δευτερόλεπτα.Στον παρακάτω πρώτο πίνακα φαίνεται η σύγκριση της κατανάλωσης μνήμης ανάμεσα στις δύο υλοποιήσεις και στον δεύτερο παρουσιάζονται τα αποτελέσματα των πειραμάτων μέτρησης της απόδοσης του EccM 2.0 στις συσκευές MICA 2.

Πίνακας 6.2: Σύγκριση της κατανάλωσης μνήμης ανάμεσα στις υλοποιήσεις EccM-1.0 και EccM-2.0.

| | EccM-1.0 (32-bit key) | EccM-2.0 (163-bit key) |
|---|---|---|
| .bss | 826B | 1,055B |
| .data | 6B | 4B |
| .text | 17,544B | 34,342B |
| stack | 976B | 81B |

Πίνακας 6.3: Απόδοση της υλοποίησης EccM-2.0.

| | Private Key Generation | Public Key Generation |
|---|---|---|
| Total Time | 0.229 sec | 34.161 sec |
| CPU Utilization | $1.690 \times 10^6$ cycles | $2.512 \times 10^8$ cycles |
| Total Energy | 0.00549 Joules | 0.816 Joules |

## 6.4 Η Εργασία TinyECC

Το 2007, οι An Liu και Peng Ning εξέδωσαν την πρώτη έκδοση της εργασίας τους με όνομα TinyECC.Το TinyECC είναι ένα πακέτο λογισμικού το οποίο παρέχει λειτουργίες κρυπτογράφησης με ελλειπτικές καμπύλες οι οποίες μπορούν εύκολα να ενσωματωθούν σε εφαρμογές ασυρμάτων δικτύων αισθητήρων.Οι λειτουργίες που παρέχει είναι οι εξής: ένα σχήμα ψηφιακής υπογραφής (ECDSA), ένα σχήμα ανταλλαγής κλειδιών (ECDH) και ένα σχήμα κρυπτογράφησης δημοσίου κλειδιού (ECIES).Επιπλεόν, παρέχει τη δυνατότητα ενεργοποίησης διαφόρων τεχνικών βελτιστοποίησης οι οποίες μπορούν να χρησιμοποιηθούν ανάλογα με τις ανάγκες της εκάστοτε εφαρμογής.Το TinyECC προορίζεται για πλατφόρμες αισθητήρων που τρέχουν το λειτουργικό σύστημα TinyOS.Είναι υλοποιημένο στη γλώσσα nesC και για βελτιστοποιήσεις αναλόγως με την πλατφόρμα στην οποία εκτελείται, κομμάτια του κώδικα είναι γραμμένα σε γλώσσα μηχανής για τις αντίστοιχες πλατφόρμες.Η αξιολόγηση της απόδοσης του TinyECC έγινε στις συσκευές MICAz, TelosB, Tmote Sky και Imote2 οι οποίες τρέχουν το λειτουργικό TinyOS.

### 6.4.1 Αρχές Σχεδίασης του TinyECC

Όπως προαναφέραμε, βασικός στόχος του TinyECC είναι να παρέχει ένα **έτοιμο για χρήση** πακέτο λογισμικού το οποίο υλοποιεί λειτουργίες κρυπτογραφίας δημοσίου κλειδιού με ελλειπτικές καμπύλες.Οι βασικές αρχές σχεδίασης του TinyECC φαίνονται παρακάτω.

- **Ασφάλεια**: Το TinyECC παρέχει σχήματα κρυπτογράφησης με ελλειπτικές καμπύλες τα οποία έχουν αποδειχθεί ασφαλή.Αυτά είναι: το σχήμα ψηφιακής υπογραφής (ECDSA), το σχήμα ανταλλαγής κλειδιών (ECDH) και το σχήμα κρυπτογράφησης δημοσίου κλειδιού (ECIES).Επιπλέον, χρησιμοποιεί τις παραμέτρους ελλειπτικών καμπυλών που προτείνονται από το SECG: Standards for Efficient Cryptography Group.

- **Μεταφερσιμότητα**: Για τη δυνατότητα εκτέλεσης του TinyECC σε διάφορες πλατφόρμες, υλοποιήθηκε στη γλώσσα nesC του λειτουργικού συστήματος TinyOS.Κάποια κομμάτια του κώδικα γράφτηκαν σε γλώσσα μηχανής για βελτιστοποιήσεις ανάλογα την πλατφόρμα εκτέλεσης.Η λειτουργικότητα του TinyECC έχει εξεταστεί με επιτυχία στις συσκευές MICAz, TelosB, Tmote Sky και Imote2.

- **Αποτελεσματικότητα**: Οι δημιουργοί του TinyECC πήραν κάποιες σχεδιαστικές αποφάσεις έτσι ώστε οι κρυπτογραφικές του λειτουργίες να εκτελούνται σωστά, γρήγορα και να καταναλώνουν λιγότερη ενέργεια.Για το στόχο αυτό υλοποίησαν την κρυπτογραφία με ελλειπτικές καμπύλες πάνω από τα πρώτα πεδία $F_p$, θεωρώντας ότι οι αριθμητικές πράξεις πάνω από τα δυαδικά πεδία $F_{2^p}$ δεν εκτελούνται αποδοτικά στους μικροελεγκτές.Επίσης, όπως αναφέραμε και προηγουμένως κάποια κρίσιμα κομμάτια του κώδικα υλοποιήθηκαν σε γλώσσα μηχανής για τις διάφορες πλατφόρμες.Με τον τρόπο αυτό κάποιες υπολογιστικά απαιτητικές αριθμητικές πράξεις εκτελούνται πολύ πιο γρήγορα.

- **Λειτουργικότητα**: Με τα κρυπτογραφικά σχήματα που παρέχει το TinyECC, δηλαδή τα ECSDA,ECIES και ECDH, καλύπτει τις βασικές ανάγκες για κρυπτογράφηση με ελλειπτικές καμπύλες.

### 6.4.2 Τεχνικές Βελτιστοποίσης του TinyECC

Για την παροχή καλύτερης απόδοσης, οι δημιουργοί του TinyECC χρησιμοποίησαν κάποιες τεχνικές βελτιστοποίησης οι οποίες παρουσιάζονται στη συνέχεια.

#### Τεχνικές Βελτιστοποίησης για Πράξεις με Μεγάλους Ακεραίους Αιθμούς

- Barrett Reduction : Ένας απλός τρόπος για την εκτέλεση modulo αναγωγής μεγάλων ακεραίων αριθμών είναι η χρήση της διαίρεσης.Η τεχνική Barrett Reduction είναι ένας ενναλακτικός τρόπος για εκτέλεση modulo αναγωγής.Η τεχνική αυτή μετατρέπει την αναγωγή ενός τυχαίου ακεραίου σε δύο πολλαπλασιασμούς και μερικές αναγωγές modulo ακεραίων

της μορφής $2^n$.Όταν χρησιμοποιείται για την αναγωγή μεγάλων ακεραίων modulo του ίδιου αριθμού πολλές φορές είναι αρκετά πιο γρήγορη από τις αναγωγές με χρήση διαίρεσης.

- Hybrid Multiplication : Οι πράξεις πολλαπλασιασμού μεγάλων ακεραίων αποθηκεύουν τους τελεστές και τα αποτελέσματα σε πίνακες.Όταν τέτοιες πράξεις υλοποιούνται σε γλώσσες υψηλού επιπέδου, ο μεταγλωττιστής δε χρησιμοποιεί αποδοτικά τους καταχωρητές του μικρο-ελεγκτή.Για το λόγο αυτό χρησιμοποιήθηκε η τεχνική Hybrid Multiplication η οποία υλοποιήθηκε σε γλώσσα μηχανής.Η τεχνική αυτή μεγιστοποιεί τη χρήση καταχωρητών και μειώνει τον αριθμό των λειτουργιών της μνήμης.

### Τεχνικές Βελτιστοποίησης για Πράξεις με Ελλειπτικές Καμπύλες

- Projective Coordinate Systems : Ο μικρο-ελεγκτής ATmega128 δεν έχει εντολή για διαίρεση με αποτέλεσμα η διαδικασία αντιστροφής ενός ακεραίου στο πεδίο της ελλειπτικής καμπύλης να είναι πιο *ακριβή* από τον πολλαπλασιασμό.Για το λόγο αυτό, η εκτέλεση των πράξεων των ελλειπτικών καμπυλών σε συντεταγμένες προβολής αντί για τις σχετικές συντεταγμένες είναι πιο αποδοτική.Οι δημιουργοί του TinyECC χρησιμοποίησαν την Jacobian αναπαράσταση για την επιτάχυνση της πρόσθεσης σημείου, του διπλασιασμού σημείου και του βαθμωτού πολλαπλασιασμού σημείου ελλειπτικής καμπύλης.

- Sliding Window Method : Η μέθοδος αυτή υλοποιήθηκε για την επιτάχυνση του βαθμωτού πολλαπλασιασμού σημείου ελλειπτικής καμπύλης.Η κλασική μέθοδος υπολογισμού του βαθμωτού πολλαπλασιασμού είναι η δυαδική.Η μέθοδος αυτή σαρώνει τα bits του αριθμού $n$ από αριστερά προς τα δεξιά, ένα bit κάθε φορά.Σε κάθε βήμα εκτελείται ένας διπλασιασμός σημείου και αναλόγως του εκάστοτε bit εκτελείται και μια πρόσθεση σημείου.Η μέθοδος Sliding Window σαρώνει $k$ bits τη φορά.Ο διπλασιασμός σημείου γίνεται $k$ φορές σε κάθε βήμα και αναλόγως τα $k$ bits εκτελείται και μια πρόσθεση σημείου.Η μέθοδος αυτή επιτυγχάνει την επιτάχυνση του βαθμωτού πολλαπλασιασμού μειώνοντας τον αριθμό των συνολικών προσθέσεων σημείου.Βέβαια, έχει μεγαλύτερες απαιτήσεις σε μνήμη.

- Shamir's Trick : Η μέθοδος αυτή χρησιμοποιείται για τη βελτιστοποίηση της ταχύτητας του σχήματος ψηφιακής υπογραφής ECDSA.

- Curve Specific Optimization : Οι περισσότερες ελλειπτικές καμπύλες που προτείνονται από το NIST και το SECG χρησιμοποιούν τους ψευδο-Mersenne πρώτους αριθμούς.Ένας ψευδο-Mersenne πρώτος αριθμός είναι

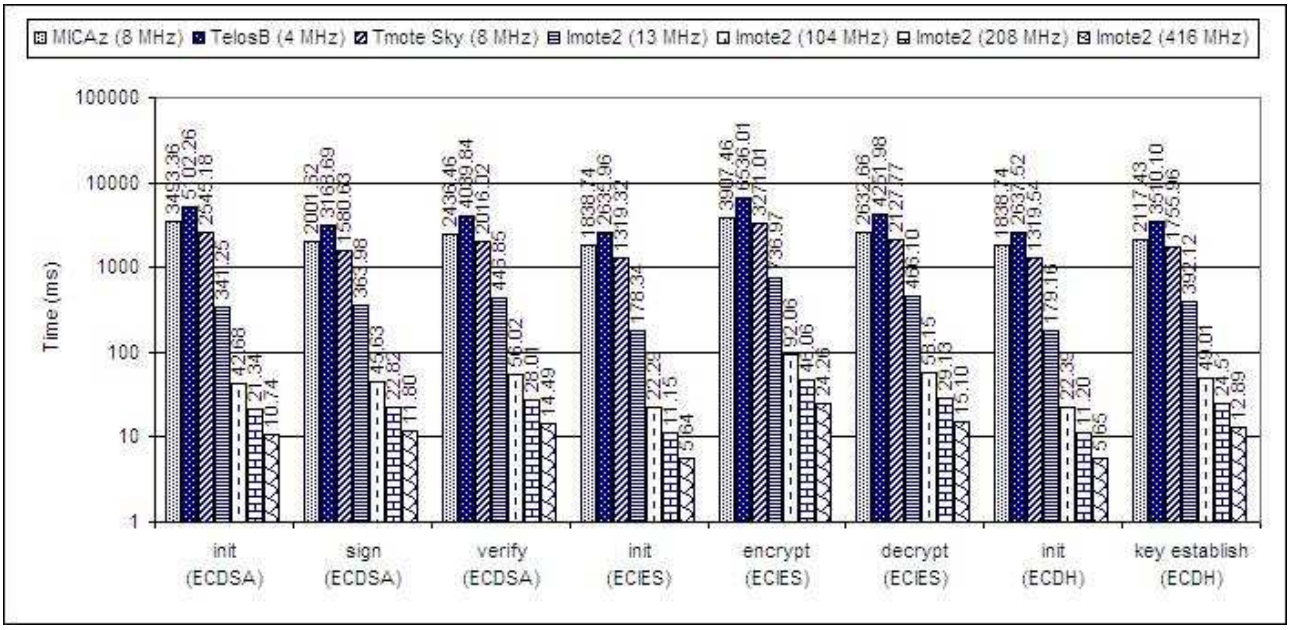
της μορφής $p = 2^n - c$, όπου $c \ll 2^n$.Η πράξη της αναγωγής modulo ενός ψευδο-Mersenne πρώτου αριθμού πραγματοποιείται με λίγους πολλαπλασιασμούς και προσθέσεις, χωρίς να γίνει χρήση διαίρεσης.Με τον τρόπο αυτό η αναγωγή γίνεται πιο γρήγορα και η χρήση τέτοιων ελλειπτικών καμπυλών βελτιώνει τη γενικότερη απόδοση του συστήματος.

### 6.4.3 Απόδοση του TinyECC

Η απόδοση της υλοποίησης TinyECC εξετάστηκε στις συσκευές MICAz (8 MHz), TelosB (4 MHz), Tmote Sky (8 MHz) και Imote2 (13, 104, 208, 416 Mhz) οι οποίες βασίζονται στο λειτουργικό σύστημα TinyOS.Χρησιμοποιείται το πρότυπο secp160r1 το οποίο προτείνεται από το SECG και ορίζει παραμέτρους για κρυπτογράφηση με ελλειπτικές καμπύλες πάνω από τα πρώτα πεδία $F_p$.Παρακάτω φαίνονται τα αποτελέσματα της απόδοσης του TinyECC σε δύο περιπτώσεις: όταν όλες οι τεχνικές βελτιστοποίησης είναι ενεργοποιημένες και όταν είναι απενεργοποιημένες.

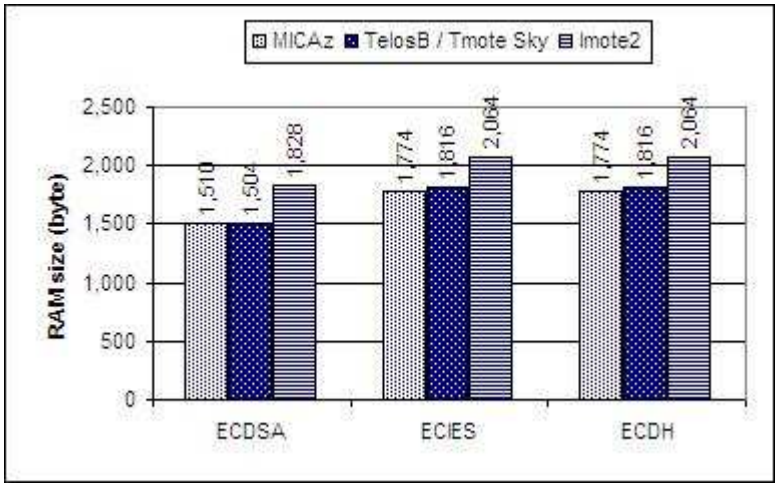
#### Ενεργοποιημένες Τεχνικές Βελτιστοποίησης

Στο παρακάτω σχήμα φαίνεται η απόδοση του TinyECC με όλες τις τεχνικές βελτιστοποίησης ενεργοποιημένες.Όπως φαίνεται σε αυτό το σχήμα οι συσκευές TelosB (4 MHz) είναι οι πιο αργές και οι συσκευές Imote2 (416 Mhz) είναι οι πιο γρήγορες.

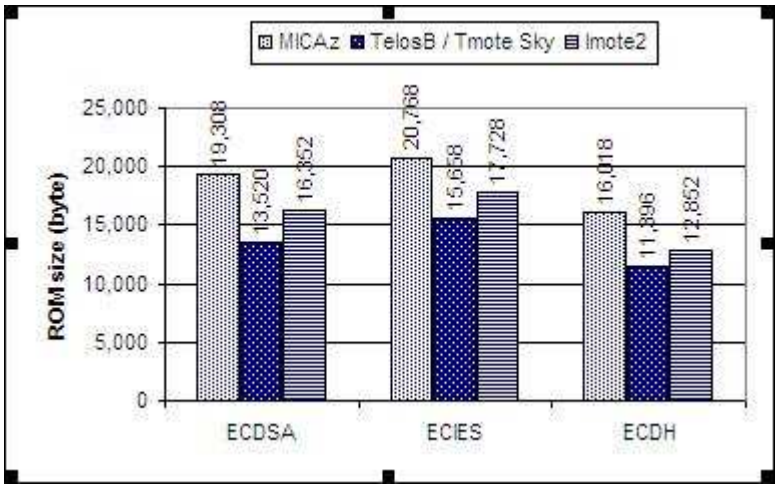



Σχήμα 6.4: Χρόνοι εκτέλεσης των λειτουργιών του TinyECC στις διάφορες πλατφόρμες όταν όλες οι τεχνικές βελτιστοποίησης είναι ενεργοποιημένες.

Στην παρακάτω εικόνα φαίνεται η κατανάλωση μνήμης του κωδικα του TinyECC.Η συσκευή Imote2 έχει τις μεγαλύτερες απαιτήσεις σε RAM λόγω των 32-bit words που χρησιμοποιεί.Η συσκευή MICAz έχει τις μεγαλύτερες απαιτήσεις σε ROM λόγω του κώδικα σε γλώσσα μηχανής που χρησιμοποιείται για βελτιστοποίηση των λειτουργιών με ελλειπτικές καμπύλες.




Σχήμα 6.5: Κατανάλωση μνήμης RAM του TinyECC όταν όλες οι τεχνικές βελτιστοποίησης είναι ενεργοποιημένες.

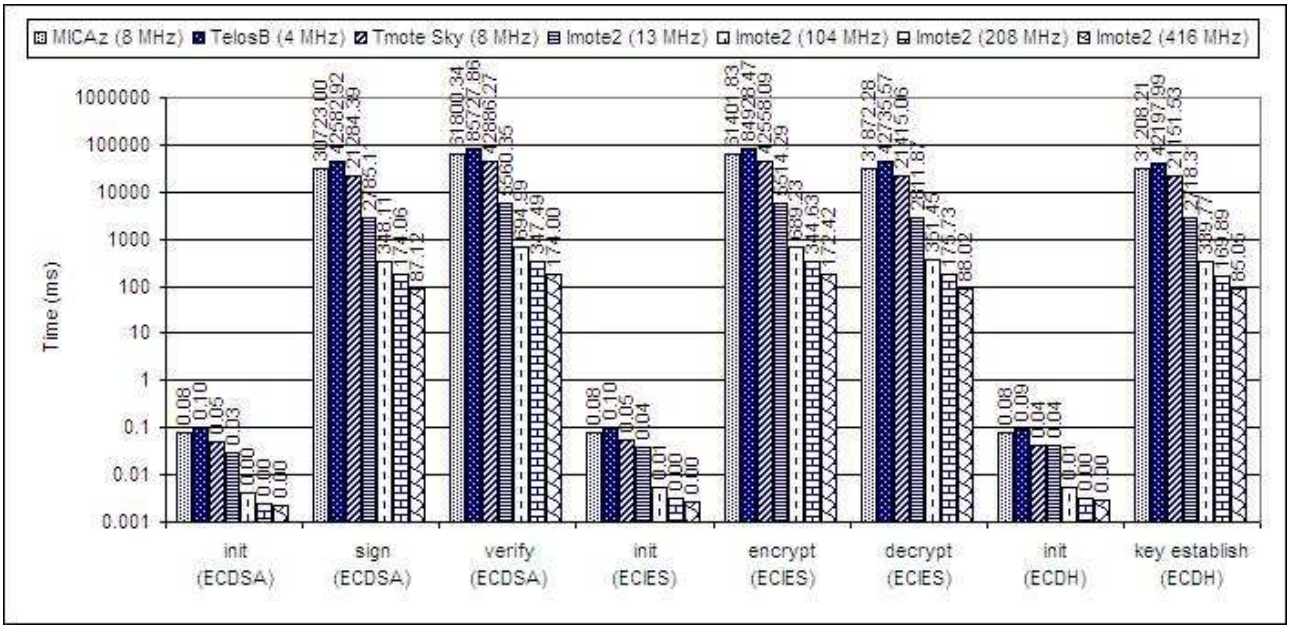



Σχήμα 6.6: Κατανάλωση μνήμης ROM του TinyECC όταν όλες οι τεχνικές βελτιστοποίησης είναι ενεργοποιημένες.

### Απενεργοποιημένες Τεχνικές Βελτιστοποίησης

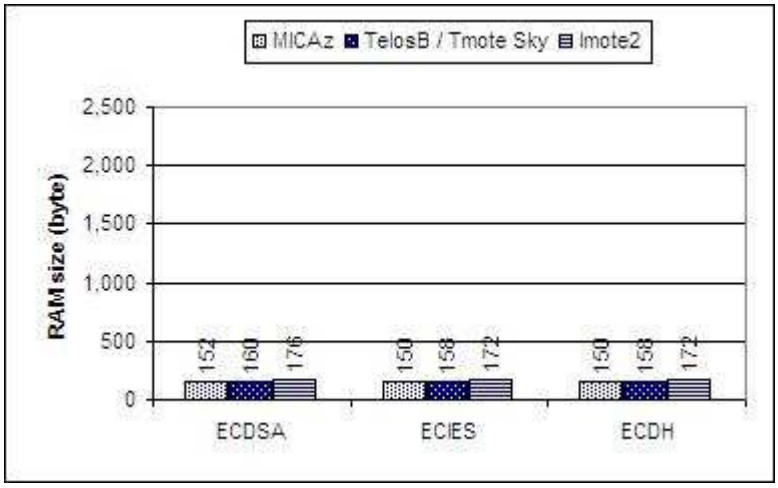
Για την ενσωμάτωση του TinyECC σε άλλες εφαρμογές του TinyOS, χρειάζεται μείωση του μεγέθους του κώδικα του.Αυτό ισχύει ειδικά στις συσκευές MICAz και TelosB που είναι χαμηλότερων πόρων σε σχέση με τις άλλες.Για το

λόγο αυτό μετρήθηκε η απόδοση του TinyECC όταν όλες οι τεχνικές βελτιστοποίησης είναι απενεργοποιημένες.Τα αποτελέσματα φαίνονται στα παρακάτω σχήματα.

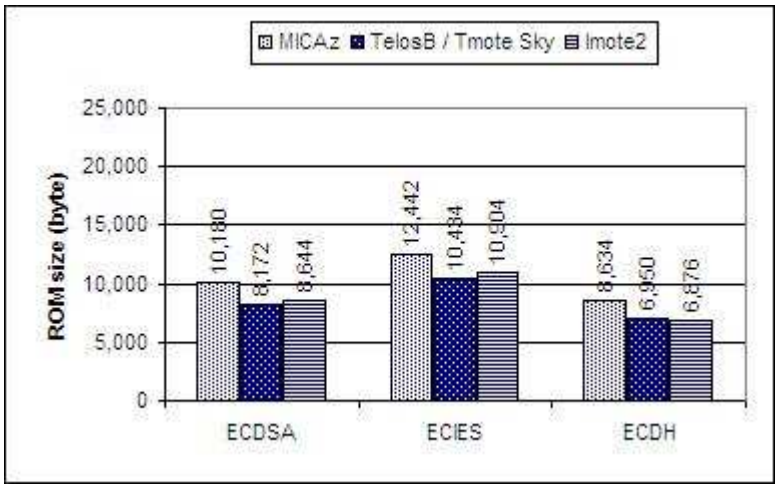



Σχήμα 6.7: Χρόνοι εκτέλεσης των λειτουργιών του TinyECC στις διάφορες πλατφόρμες όταν όλες οι τεχνικές βελτιστοποίησης είναι απενεργοποιημένες.

Όπως παρατηρούμε από την παραπάνω γραφική παράσταση οι χρόνοι εκτέλεσης των κρυπτογραφικών λειτουργιών του TinyECC αυξάνονται αρκετά όταν οι τεχνικές βελτιστοποίησης είναι απενεργοποιημένες.Το μέγεθος του κώδικα, σε αντίθεση, μειώνεται δραματικά όπως φαίνεται στις παρακάτω γραφικές παραστάσεις.Για παράδειγμα, οι απαιτήσεις σε RAM για τις συσκευές MICAz είναι περίπου 150 bytes και το μέγεθος ROM του ECIES μειώνεται από τα 20,768 bytes στα 12,442 bytes ($\simeq 40\%$).




Σχήμα 6.8: Κατανάλωση μνήμης RAM του TinyECC όταν όλες οι τεχνικές βελτιστοποίησης είναι ενεργοποιημένες.




Σχήμα 6.9: Κατανάλωση μνήμης ROM του TinyECC όταν όλες οι τεχνικές βελτιστοποίησης είναι ενεργοποιημένες.

## 6.5 Περιγραφή του Πρωτοκόλλου μας

Όπως αναφέραμε και προηγουμένως, η εργασία μας βασίστηκε στην υλοποίηση υποδομής δημοσίου κλειδιού για διανομή κλειδιών με ελλειπτικές καμπύλες του David Malan.Από την εργασία αυτή, μεταφέραμε στην πλατφόρμα

iSense, κάνοντας τις απαραίτητες αλλαγές (π.χ. data types), τις συναρτήσεις που σχετίζονται με πράξεις μεγάλων ακεραίων (big interger routines), τις συναρτήσεις πεδίων (field routines) καθώς και τις συναρτήσεις για πράξεις πάνω σε ελλειπτική καμπύλη όπως πρόσθεση σημείων και πολλαπλασιασμό σημείου με ακέραιο αριθμό.Οι συναρτήσεις αυτές, αποτέλεσαν τη βάση για να υλοποιήσουμε κατάλληλους μηχανισμούς κρυπτογράφησης και αποκρυπτογράφησης μπλοκ δεδομένων καθώς και ένα πρωτόκολλο ασφαλούς επικοινωνίας, το εξετάστηκε σε ένα δίκτυο χαμηλών πόρων αποτελούμενο από δύο συσκευές iSense.Στην αρχή του πρωτοκόλλου, οι κόμβοι φορτώνουν στη μνήμη τους τις παραμέτρους για κρυπτογράφηση με ελλειπτικές καμπύλες (σημείο βάσης, εξίσωση καμπύλης κτλ).Στη συνέχεια, δημιουργούν τα ιδιωτικά και δημόσια κλειδιά τους και κατόπιν εκτελούν το σχήμα συμφωνίας κλειδιού Elliptic Curve Diffie Hellman.Έτσι δημιουργούν ένα κοινό διαμοιραζόμενο μυστικό το οποίο είναι ένα νέο σημείο πάνω στην καμπύλη.Τέλος, ανταλλάσουν μπλοκ δεδομένων κρυπτογραφημένα με το κοινό διαμοιραζόμενο μυστικό, τα οποία αντίστοιχα αποκρυπτογραφούν.Η απόδοση του πρωτοκόλλου που αναπτύξαμε, συγκρίθηκε ως προς τη χρονική καθυστέρηση με τις αντίστοιχες εργασίες του Malan και TinyECC.Αναλυτικά, το πρωτόκολλο περιγράφεται στη συνέχεια.

### 6.5.1 Φόρτωση της Ελλειπτικής Καμπύλης και του Σημείου Βάσης στους Κόμβους

Όπως και στην εργασία του David Malan, έτσι και στη δικιά μας υπάρχει μια δομή (C struct) που ονομάζεται Params και περιέχει τις κατάλληλες μεταβλητές για τον ορισμό των παραμέτρων της κρυπτρογράφησης ελλειπτικών καμπυλών.Οι παράμετροι αυτές είναι η εξίσωση της καμπύλης, η τάξη της, το απλοποιημένο πολυώνυμο και το σημείο βάσης $G$.Μία κατάλληλη συνάρτηση που ονομάζεται **init()** αρχικοποιεί την ελλειπτική καμπύλη και τις παραμέτρους της σε κάθε κόμβο.Έτσι, η εξίσωση της καμπύλης είναι η

$$y^2 + xy = x^3 + x^2 + 1$$

σαν απλοποιημένο πολυώνυμο χρησιμοποιείται το:

$$f(x) = x^{163} + x^7 + x^6 + x^3 + 1$$

η τάξη της καμπύλης ( ο αριθμός των σημείων πάνω σε αυτή) είναι:

$$0x4000000000000000000020108a2e0cc0d99f8a5ef$$

και το σημείο βάσης είναι το $G = (G_x, G_y)$ με :

$$G_x = 0x2fe13c0537bbc11acaa07d793de4e6d5e5c94eee8$$

και

$$G_y = 0x289070fb05d38ff58321f2e800536d538ccdaa3d9$$

### 6.5.2 Δημιουργία Ιδιωτικού Κλειδιού

Κάθε κόμβος δημιουργεί έναν τυχαίο αριθμό (μήκους 21 bytes) ο οποίος αποτελεί το ιδιωτικό του κλειδί.Η κατάλληλη δομή PrivKey αρχικοποιείται με κλήση της συνάρτησης **gen_private_key()**.Η συνάρτηση αυτή χρησιμοποιεί τις συναρτήσεις srand() και rand() που παρέχει η κλάση PseudoRandomNumberGenerator της πλατφόρμας iSense και επιστρέφει το ιδιωτικό κλειδί του κάθε κόμβου.Για τη γέννηση ψευδοτυχαίων αριθμών, η συνάρτηση isense:: PseudoRandomNumberGenerator:: rand δέχεται μια αρχικοποιημένη seed τιμή από τη συνάρτηση isense:: PseudoRandomNumberGenerator:: srand(), η οποία καλείται με όρισμα την εκάστοτε ταυτότητα (os().id()) του κάθε κόμβου.

### 6.5.3 Πρόσθεση, Βαθμωτός Πολλαπλασιασμός Σημείων Ελλειπτικής Καμπύλης και Δημιουργία Δημοσίου Κλειδιού

Για την πρόσθεση δύο σημείων πάνω στην ελλειπτική καμπύλη χρησιμοποιείται η συνάρτηση c_add() η οποία δέχεται ως ορίσματα δύο σημεία της καμπύλης και επιστρέφει το αποτέλεσμα της πρόσθεσης.Για τον πολλαπλασιασμό ενός σημείου της ελλειπτικής καμπύλης με έναν ακέραιο αριθμό χρησιμοποιείται η συνάρτηση c_mul() η οποία δέχεται για ορίσματα ένα σημείο της καμπύλης και έναν αριθμό και επιστρέφει το νέο σημείο πάνω στην καμπύλη.Η συνάρτηση αυτή υλοποιεί τη δυαδική μέθοδο για τον βαθμωτό πολλαπλασιασμό σημείου ελλειπτικής καμπύλης.Με βάση τη συνάρτηση c_mul() δημιουργήσαμε τη συνάρτηση **gen_public_key()** η οποία δημιουργεί το δημόσιο κλειδί του κάθε κόμβου.Αυτό γίνεται πολλαπλασιάζοντας το ιδιωτικό του κλειδί του με το σημείο βάσης $G$ της καμπύλης που έχει αρχικοποιηθεί από τη συνάρτηση init().Αντιστοίχως, λειτουργεί και η συνάρτηση **gen_shared_secret()** η οποία δημιουργεί το κοινό διαμοιραζόμενο μυστικό, δηλαδή ένα σημείο πάνω στην καμπύλη, ανάμεσα σε δύο κόμβους.Το σημείο αυτό προκύπτει πολλαπλασιάζοντας το ιδιωτικό κλειδί ενός κόμβου με το αντίστοιχο δημόσιο κλειδί του κόμβου με τον οποίο επιθυμεί να επικοινωνήσει.

### 6.5.4 Elliptic Curve Diffie-Hellman

Όπως αναφέραμε και προηγουμένως, στην αρχή του πρωτοκόλλου μας εκτελείται η διαδικασία συμφωνίας κλειδιού Diffie-Hellman έτσι ώστε οι δύο συσκευές του δικτύου να συμφωνήσουν σε ένα κοινό μυστικό.Το μυστικό αυτό είναι ένα νέο σημείο πάνω στην ελλειπτική καμπύλη.Κατά την διαδικασία αυτή, οι κόμβοι υπολογίζουν το προσωπικό και δημόσιο κλειδί τους με τη χρήση των συναρτήσεων gen_private_key() και gen_public_key().Στη συνέχεια ανταλλάσουν

μηνύματα που περιέχουν τα δημόσια κλειδιά τους με χρήση της συνάρτησης send() που παρέχει η κλάση Radio της πλατφόρμας iSense.Η παραλαβή των μηνυμάτων γίνεται με χρήση της συνάρτησης receive() που παρέχει η κλάση Receiver της πλατφόρμας iSense.Αφού η κάθε συσκευή παραλάβει το δημόσιο κλειδί της άλλης, υπολογίζει το κοινό διαμοιραζόμενο μυστικό πολλαπλασιάζοντας το ιδιωτικό της κλειδί με το κλειδί που έλαβε.Αυτό γίνεται με τη βοήθεια της συνάρτησης gen_shared_secret().Μετά από τη διαδικασία αυτή, οι κόμβοι έχουν στην κατοχή τους ένα κοινό κλειδί με βάση το οποίο μπορούν να κρυπτογραφούν και να αποκρυπτογραφούν τα μηνύματα που επιθυμούν να ανταλλάξουν.

### 6.5.5 Μηχανισμοί Κρυπτογράφησης και Αποκρυπτογράφησης Δεδομένων

Στα πλαίσια αυτής της εργασίας αναπτύξαμε τρεις διαφορετικούς μηχανισμούς κρυπτογράφησης/αποκρυπτογράφησης δεδομένων οι οποίοι παρουσιάζονται παρακάτω.

#### Provably Secure Elliptic Curve Encryption Scheme

Ο πρώτος μηχανισμός βασίζεται στο σχήμα PSEC : Provably Secure Elliptic Curve Encryption Scheme που παρουσιάζεται στο πρότυπο IEEE P1363 για κρυπτογραφία δημοσίου κλειδιού.Αφού έχει γίνει συμφωνία των παραμέτρων της κρυπτογράφησης ελλειπτικών καμπυλών(εξίσωση καμπύλης, σημείο βάσης $G$ κτλ) και μιας συνάρτησης MAC(Message Authentication Code), ο χρήστης $A$ κρυπτογραφεί το μήνυμα $M$ μήκους $length$ με το δημόσιο κλειδί $Q$ του χρήστη $B$ ως εξής:

1. Υπολογίζει το ιδιωτικό του κλειδί $k$ και κατόπιν το δημόσιο του κλειδί $R = kG = (x_R, y_R)$.

2. Υπολογίζει το κοινό διαμοιραζόμενο μυστικό $P$ πολλαπλασιάζοντας το ιδιωτικό του κλειδί $k$ με το δημόσιο κλειδί $Q$ του χρήστη $B$.Δηλαδή, $P = kQ = (x_P, y_P)$.

3. Χρησιμοποιεί τη $x$ συντεταγμένη του κοινού μυστικού με μια συνάρτηση παραγωγής κλειδιού (KDF: Key Derivation Function) έτσι ώστε να παράγει ένα κλειδί $K$ μήκους $length + 20$.Τα πρώτα $length$ bytes του κλειδιού $K$ χρησιμοποιούνται ως κλειδί κρυπτογράφησης $EK$ και τα υπόλοιπα 20 bytes χρησιμοποιούνται ως κλειδί αυθεντικότητας $MK$(MAC: Message Authentication Code).

4. Ανάμεσα στο μήνυμα $M$ και το κλειδί κρυπτογράφησης $EK$ εκτελείται η λειτουργία XOR, δηλαδή $EncData = M \oplus EK$.

5. Στη συνέχεια χρησιμοποιείται ένα σχήμα MAC πάνω στα κρυπτογραφημένα δεδομένα και το κλειδί $MK$ έτσι ώστε να παραχθεί ένα tag D για το μήνυμα.

6. Η έξοδος του αλγόριθμου κρυπτογράφησης είναι η συνένωση του δημοσίου κλειδιού $R$ με τα κρυπτογραφημένα δεδομένα $EncData$ και το tag D που χρησιμοποιείται για αυθεντικοποίηση.Δηλαδή, $C = R \| EncData \| D$.

Στο σημείο αυτό αναφέρουμε ότι ως σχήμα MAC (Message Authentication Code) χρησιμοποιήσαμε αυτό που παρουσιάζεται στο πρότυπο RFC 2104.Το σχήμα αυτό χρησιμοποιεί τον αλγόριθμο SHA-1 (Secure Hashing Algorithm) ως συνάρτηση κατακερματισμού.Ο SHA-1 επιστρέφει ένα 160-bit hash για κάθε μηνύμα μέγιστου μεγέθους $(2^{64} - 1)bits$.

Η αντίστοιχη διαδικασία αποκρυπτογράφησης του κρυπτογραφημένου μηνύματος $C$ από τον χρήστη $B$ περιγράφεται παρακάτω.

1. Αρχικά, από το κρυπτογραφημένο μήνυμα διαβάζει το σημείο $R$ που είναι το δημόσιο κλειδί του χρήστη $A$.Κατόπιν, υπολογίζει το κοινό διαμοιραζόμενο μυστικό $P$ πολλαπλασιάζοντας το ιδιωτικό του κλειδί $q$ με το κλειδί $R$ του χρήστη $A$.Δηλαδή, $P = qR = (x_P, y_P)$.

2. Χρησιμοποιεί τη $x$ συντεταγμένη του κοινού μυστικού με τη συνάρτηση KDF και παράγει ένα κλειδί $K$ μήκους $length + 20$.Τα πρώτα $length$ bytes του κλειδιού $K$ χρησιμοποιούνται ως κλειδί αποκρυπτογράφησης $DK$ και τα υπόλοιπα 20 bytes χρησιμοποιούνται ως κλειδί αυθεντικότητας $MK$(MAC: Message Authentication Code).

3. Χρησιμοποιεί το σχήμα MAC πάνω στο κλειδί $MK$ και τα κρυπτογραφημένα δεδομένα έτσι ώστε να επιβεβαιώσει το tag D.Αν το αποτέλεσμα της συνάρτησης MAC δεν είναι ίδιο με το tag απορίπτει το μήνυμα.Με τον τρόπο αυτό, ο χρήστης $B$ είναι σίγουρος ότι τα κρυπτογραφημένα δεδομένα που έλαβε, δημιουργήθηκαν από το χρήστη $A$.

4. Αφού επιβεβαιώσει το tag D εκτελεί τη λειτουργία XOR ανάμεσα στα κρυπτογραφημένα δεδομένα και το κλειδί αποκρυπτογράφησης $DK$ και λαμβάνει το αρχικό μήνυμα $M$.

Οι συναρτήσεις που υλοποιούν αυτόν το μηχανισμό είναι οι **psec_encrypt()** και **psec_decrypt()**.

### Elliptic Curve Encryption Scheme

Ο δεύτερος μηχανισμός κρυπτογράφησης βασίζεται στον αλγόριθμο ECES: Elliptic Curve Encryption Scheme.Αφού έχει γίνει συμφωνία των παραμέτρων της κρυπτογράφησης ελλειπτικών καμπυλών(εξίσωση καμπύλης, σημείο βάσης $G$ κτλ), ο χρήστης $A$ κρυπτογραφεί το μήνυμα $M$ μήκους $length$ με το δημόσιο κλειδί $Q$ του χρήστη $B$ ως εξής:

1. Υπολογίζει το ιδιωτικό του κλειδί $k$ και κατόπιν το δημόσιο του κλειδί $R = kG = (x_R, y_R)$.

2. Υπολογίζει το κοινό διαμοιραζόμενο μυστικό $P$ πολλαπλασιάζοντας το ιδιωτικό του κλειδί $k$ με το δημόσιο κλειδί $Q$ του χρήστη $B$.Δηλαδή, $P = kQ = (x_P, y_P)$.

3. Πολλαπλασιάζει το μήνυμα $M$ με τη $x$ συντεταγμένη του κοινού διαμοιραζόμενου μυστικου.Δηλαδή, $EncData = Mx_P(modp)$, όπου $p$ η τάξη της καμπύλης.Πρέπει να αναφέρουμε στο σημείο αυτό ότι ο πολλαπλασιασμός γίνεται πάνω από το πεδίο της ελλειπτικής καμπύλης και για την εκτέλεσή του χρησιμοποιείται η κατάλληλη συνάρτηση f_mul().

4. Η έξοδος του αλγόριθμου κρυπτογράφησης είναι η συνένωση του δημοσίου κλειδιού $R$ με τα κρυπτογραφημένα δεδομένα $EncData$ .Δηλαδή, $C = R||EncData$.

Για την αποκρυπτογράφηση του μηνύματος $C$, ο παραλήπτης $B$ ακολουθεί τα εξής βήματα:

1. Αρχικά, από το κρυπτογραφημένο μήνυμα διαβάζει το σημείο $R$ που είναι το δημόσιο κλειδί του χρήστη $A$.Κατόπιν, υπολογίζει το κοινό διαμοιραζόμενο μυστικό $P$ πολλαπλασιάζοντας το ιδιωτικό του κλειδί $q$ με το κλειδί $R$ του χρήστη $A$.Δηλαδή, $P = qR = (x_P, y_P)$.

2. Αντιστρέφει την $x$ συντεταγμένη του κοινού μυστικού στο πεδίο της ελλειπτικής καμπύλης με χρήση της κατάλληλης συνάρτησης f_inv().

3. Πολλαπλασιάζει το κρυπτογραφημένο μήνυμα με την ανεστραμένη συντεταγμένη $x$ και λαμβάνει το αρχικό μήνυμα.

Οι συναρτήσεις που υλοποιούν αυτόν το μηχανισμό είναι οι **eces_encrypt()** και **eces_decrypt()**.

### Elliptic Curve El Gamal

Τέλος, έγινε υλοποίηση και του αντίστοιχου μηχανισμού κρυπτογράφησης El Gamal για ελλειπτικές καμπύλες.Στο σχήμα αυτό, ο αποστολέας έχει κωδικοποιήσει το μήνυμα του σε κάποιο σημείο που περιέχεται στο πεδίο της ελλειπτικής καμπύλης.Για την κρυπτογράφηση του σημείου $M$ ο αποστολέας $A$ ακολουθεί την παρακάτω διαδικασία.

1. Υπολογίζει το ιδιωτικό του κλειδί $k$ και κατόπιν το δημόσιο του κλειδί $R = kG = (x_R, y_R)$.

2. Υπολογίζει το κοινό διαμοιραζόμενο μυστικό $P$ πολλαπλασιάζοντας το ιδιωτικό του κλειδί $k$ με το δημόσιο κλειδί $Q$ του χρήστη $B$.Δηλαδή, $P = kQ = (x_P, y_P)$.

3. Προσθέτει το σημείο $M$ με το κοινό διαμοιραζόμενο μυστικό, δηλαδή $EncPoint = P + M$.Για την πρόσθεση του σημείου $M$ με το κοινό διαμοιραζόμενο μυστικό πάνω στην καμπύλη χρησιμοποιείται η κατάλληλη συνάρτηση c_add().

4. Η έξοδος του αλγόριθμου κρυπτογράφησης είναι ένα ζευγάρι σημείων της καμπύλης.Το ζευγάρι αποτελείται από το δημόσιο κλειδί $R$ του χρήστη $A$ και τα κρυπτογραφημένο σημείο $EncPoint$ .Δηλαδή, $C = (R, EncPoint)$.

Ο παραλήπτης αντίστοιχα ακολουθεί την παρακάτω διαδικασία για την αποκρυπτογράφηση του σημείου $M$ που του έστειλε ο $A$.

1. Αρχικά, από το κρυπτογραφημένο μήνυμα διαβάζει το σημείο $R$ που είναι το δημόσιο κλειδί του χρήστη $A$ και υπολογίζει το κοινό διαμοιραζόμενο μυστικό $P$, πολλαπλασιάζοντας το ιδιωτικό του κλειδί $q$ με το κλειδί $R$ του χρήστη $A$.Δηλαδή, $P = qR = (x_P, y_P)$.

2. Αφαιρεί από το κρυπτογραφημένο σημείο $EncPoint$ το κοινό διαμοιραζόμενο μυστικό και λαμβάνει το αρχικό σημείο που επιθυμούσε να του στείλει ο αποστολέας $A$.Για την αφαίρεση του κοινού διαμοιραζόμενου μυστικού από το κρυπτογραφημένο μήνυμα, υπολογίζεται το αντίθετο σημείο του κοινού διαμοιραζόμενου μυστικού πάνω στην καμπύλη και προστίθεται στο κρυπτογραφημένο μήνυμα.

Οι συναρτήσεις που υλοποιούν τον παραπάνω μηχανισμό είναι οι **elgamal_encrypt()** και **elgamal_decrypt()**.Αναφέρουμε στο σημείο αυτό ότι το σχήμα κρυπτογράφησης El Gamal μπορεί να χρησιμοποιηθεί και για την ασφαλή μεταφορά κλειδιών ανάμεσα στους κόμβους.

## 6.6 Απόδοση του Πρωτοκόλλου και Συμπεράσματα

Παρακάτω εξετάζουμε την απόδοση του πρωτοκόλλου μας ως προς την χρονική καθυστέρηση και τη συγκρίνουμε με προηγούμενες εργασίες.

### 6.6.1 Απόδοση του Πρωτοκόλλου

Για την μέτρηση της απόδοσης της κρυπτογράφησης με ελλειπτικές καμπύλες υπολογίσαμε τις χρονικές διάρκειες γέννησης ιδιωτικού/δημόσιου κλειδιού, κρυπτογράφησης/αποκρυπτογράφησης δεδομένων και της διαδικασίας συμφωνίας κλειδιού Diffie-Hellman στις συσκευές iSense.Οι χρόνοι αυτοί μετρήθηκαν σαν μέσος όρος 100 επαναλήψεων της κάθε λειτουργίας και φαίνονται στον παρακάτω πίνακα.Όπως είδαμε από τα παραπάνω σχήματα κρυπτογράφησης/αποκρυπτογράφησης, η κρυπτογράφηση δεδομένων προυποθέτει δυο πολλαπλασιασμούς πάνω στην καμπύλη ενώ η αποκρυπτογράφηση μόνο ένα.Επίσης, ο χρόνος της διαδικασίας ανταλλαγής κλειδιών Diffie-Hellman μετρήθηκε σαν το χρόνο δημιουργίας των ιδιωτικών/δημοσίων κλειδιών των κόμβων, συν το χρόνο της αποστολής των μηνυμάτων που περιέχουν τα δημόδια κλειδιά, συν το χρόνο δημιουργίας του κοινού διαμοιραζόμενου κλειδιού στην κάθε συσκευή.

Πίνακας 6.4: Χρόνος εκτέλεσης σε sec των διάφορων λειτουργιών στις συσκευές iSense.

| Λειτουργία | Χρόνος |
|---|---|
| Private Key Generation | 0.0924 sec |
| Public Key Generation | 11.6941 sec |
| ECES Encrypt | 22.5482 sec |
| ECES Decrypt | 11.9986 sec |
| PSEC Encrypt | 22.9369 sec |
| PSEC Decrypt | 11.8483 sec |
| El Gamal Encrypt | 22.6698 sec |
| El Gamal Decrypt | 12.1214 sec |
| Elliptic Curve Diffie-Hellman Key Agreement | 24.6644 sec |

Από τις παραπάνω μετρήσεις παρατηρούμε ότι η διαδικασία δημιουργίας ενός δημοσίου κλειδιού είναι αρκετά χρονοβόρα σε σχέση με τη διαδικασία δημιουργίας ενός ιδιωτικού κλειδιού.Αυτό συμβαίνει διότι η δημιουργία ενός δημοσίου κλειδιού βασίζεται στον πολλαπλασιασμό ενός σημείου της ελλειπτικής καμπύλης με έναν μεγάλο ακέραιο αριθμό που αποτελεί την πιο απαιτητική

διαδικασία στην κρυπτογραφία με ελλειπτικές καμπύλες.Αυτό φαίνεται και από τις μετρήσεις για τα σχήματα κρυπτογράφησης/αποκρυπτογράφησης αφού ως χρόνος κρυπτογράφησης μετράται ο χρόνος δημιουργίας του δημοσίου και του κοινού κλειδιού (πολλαπλασιασμός σημείου καμπύλης με μεγάλο αριθμό) συν αυτόν της πραγματικής διαδικασίας.Το σχήμα SPEC είναι το πιο αργό καθώς περιλαμβάνει και τη διαδικασία παραγωγής του MAC στο κάθε μήνυμα που κρυπτογραφείται.Παρόλα αυτά, οι χρονικές διαφορές μεταξύ των σχημάτων κρυπτογράφησης/αποκρυπτογράφησης είναι πολύ μικρές.Τέλος, παρατηρούμε ότι η χρονική διάρκεια εκτέλεσης του πρωτοκόλλου Elliptic Curve Diffie-Hellman είναι περίπου 24.6 sec.Στη χρονική αυτή διάρκεια περιλαμβάνονται δύο πολλαπλασιασμοί σημείου ελλειπτικής καμπύλης με μεγάλο αριθμό, ένας για τη δημιουργία του δημοσίου κλειδιού και ένας για τη δημιουργία του κοινού κλειδιού, καθώς και οι χρόνοι ανταλλαγής των μηνυμάτων που περιλαμβάνουν τα δημόσια κλειδιά (μέγεθος μηνυμάτων 43 bytes).Από το γεγονός αυτό συμπεραίνουμε ότι η ανταλλαγή μηνυμάτων στις συσκευές iSense είναι πολύ γρήγορη και ότι μετά από υπολογισμούς 24 δευτερολέπτων δύο συσκευές είναι έτοιμες για ανταλλαγή κρυπτογραφημένων μηνυμάτων, ενδεχομένως με κάποιο γρήγορο συμμετρικό αλγόριθμο κρυπτογράφησης.

Με χρήση της διεπαφής iShell μετρήσαμε την κατανάλωση μνήμης του μεταφρασμένου κώδικα πρωτοκόλλου μας στο μικρο-ελεγκτή JENNIC 5139R1 της συσκευής iSense.Τα αποτελέσματα φαίνονται στον παρακάτω πίνακα.

Πίνακας 6.5: Κατανάλωση μνήμης της υλοποιησής μας στη συσκευή iSense.

| iSense JENNIC 5139R1 | Κατανάλωση Μνήμης |
|---|---|
| Total Size | 80200B |
| .bss | 488B |
| .data | 124B |
| .text | 79588B |

Παρατηρούμε ότι το συνολικό μέγεθος της μνήμης που καταναλώνει το πρωτόκολλο μας στη συσκευή iSense είναι περίπου 80Kb.Γενικά, οι κατασκευαστές της πλατφόρμας iSense προτείνουν ότι ο κώδικας των εφαρμογών που πρόκειται να φορτωθεί στις συσκευές πρέπει να είναι μικρότερος των 90Kb.

### 6.6.2 Σύγκριση του Πρωτοκόλλου μας με Προηγούμενες Εργασίες

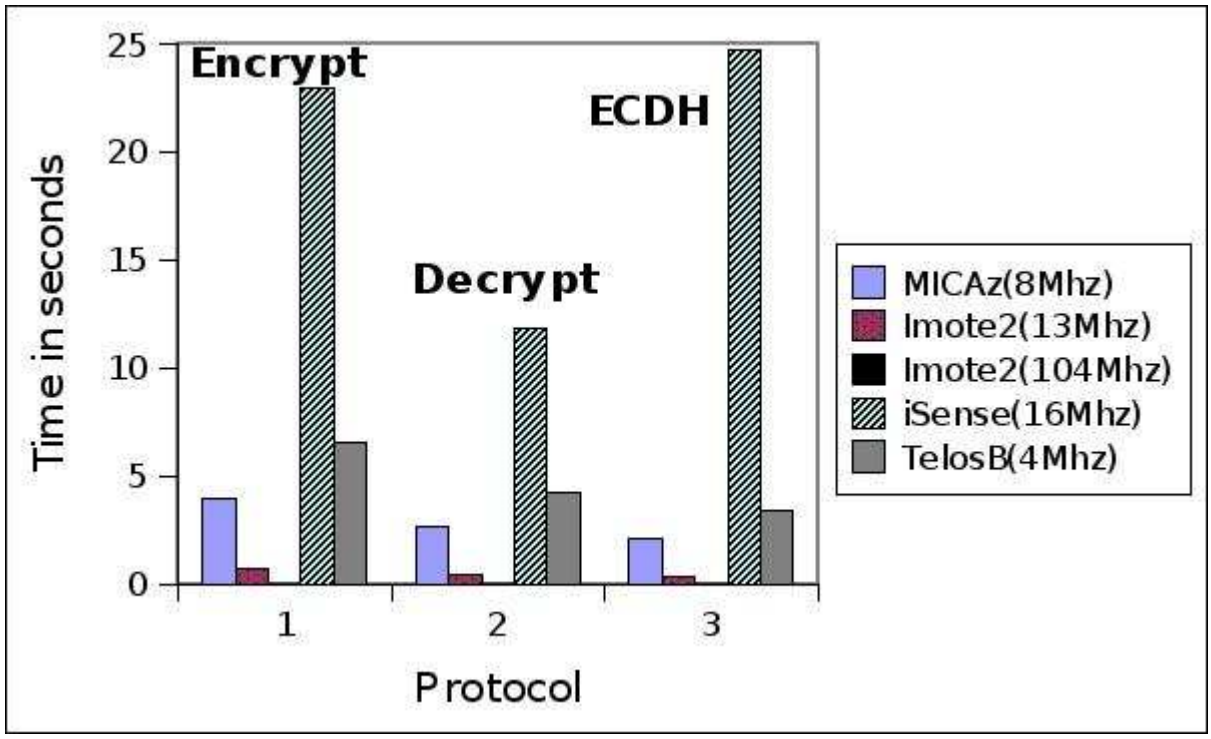
Αρχικά, συγκρίνουμε την απόδοση του πρωτοκόλλου μας με το αντίστοιχο πρωτόκολλο του Malan.Αναφέραμε παραπάνω ότι στην εργασία του Malan η γέννηση ενός ιδιωτικού κλειδιού εκτελείται σε 0.229 δευτερόλεπτα και η γέννη-

ση ενός δημοσίου κλειδιού σε 34.116 δευτερόλεπτα.Το πρωτόκολλό μας στις αντίστοιχες λειτουργίες είναι αρκετά πιο γρήγορο όπως φαίνεται στον παρακάτω πίνακα.Το γεγονός αυτό οφείλεται κυρίως στον δυνατότερο και ταχύτερο μικρο-ελεγκτή που έχει η συσκευή iSense.Θυμίζουμε ότι ο μικρο-ελεγκτής της συσκευής MICA2 λειτουργεί στα 7.38 MHz ενώ ο μικρο-ελεγκτής JENNIC 5139R1 της συσκευής iSense λειτουργεί στα 16Mhz.Η ταχύτερη γέννηση ενός δημοσίου κλειδιού στη συσκευή iSense συμαίνει ότι ο βαθμωτός πολλαπλασιασμός σημείου εκτελείται πολύ πιο γρήγορα, άρα και το πρωτόκολλο συμφωνίας κλειδιών ECDH εκτελείται πολύ πιο γρήγορα.

Πίνακας 6.6: Σύγκριση των λειτουργιών γέννησης ιδιωτικού και δημοσίου κλειδιού ανάμεσα στις υλοποιήσεις EccM-2.0 του Malan στις συσκευές MICA2 και της δικιάς μας στις συσκευές iSense.

| Λειτουργία | MICA2 | iSense |
|---|---|---|
| Private Key Generation | 0.229 sec | 0.092 sec |
| Public Key Generation | 34.161 sec | 11.694 sec |

Εν συνεχεία, συγκρίναμε τα αποτελέσματα του πρωτοκόλλου μας με αυτά της εργασίας TinyECC.Στην εργασία TinyECC έχει υλοποιηθεί επίσης ο αλγόριθμος PSEC για τον οποίο συγκρίνουμε τους χρόνους εκτέλεσης κρυπτογράφησης/αποκρυπτογράφησης.Στο πρώτο σχήμα φαίνονται οι χρόνοι εκτέλεσης των λειτουργιών Encrypt, Decrypt και ECDH του TinyECC και του πρωτοκόλλου μας.Τονίζουμε ότι οι χρόνοι εκτέλεσης του TinyECC φαίνονται για διάφορες πλατφόρμες στις οποίες εκτελείται και ότι όλες οι τεχνικές βελτιστοποίησης είναι ενεργοποιημένες.Όπως φαίνεται και στο παρακάτω σχήμα οι χρόνοι εκτέλεσης του TinyECC είναι σαφώς γρηγορότεροι.Αυτό οφείλεται τόσο στις τεχνικές βελτιστοποίησης όσο και στους γρηγορότερους μικρο-ελεγκτές που παρέχουν κάποιες από τις πλατφόρμες που εκτελείται το TinyECC.Ακόμα, οι δημιουργοί του TinyECC επέλεξαν την ανάπτυξη κρυπτογραφίας με ελλειπτικές καμπύλες πάνω από τα πρώτα πεδία $F_p$ με την αιτιολογία ότι οι αριθμητικές πράξεις πάνω από τα δυαδικά πεδία $F_{2^p}$ (στα οποία βασίζεται η υλοποίηση μας) δεν εκτελούνται αποδοτικά στους μικρο-ελεγκτές χαμηλών πόρων των αισθητήρων.Για παράδειγμα, ενώ οι συσκευές TelosB(4MHz) ,MICAz (8MHz) και Imote2 (13Mhz) είναι πιο αργές από τη συσκευή iSense (16Mhz) πετυχαίνουν καλύτερους χρόνους κυρίως λόγω της τεχνικής Sliding Window για το βαθμωτό πολλαπλασιασμό σημείου ελλειπτικής καμπύλης.Το πρωτόκολλο μας, χρησιμοποιεί τη δυαδική μέθοδο για το βαθμωτό πολλαπλασιασμό και για το λόγο αυτό η διαδικασία αυτή είναι αρκετά πιο αργή.

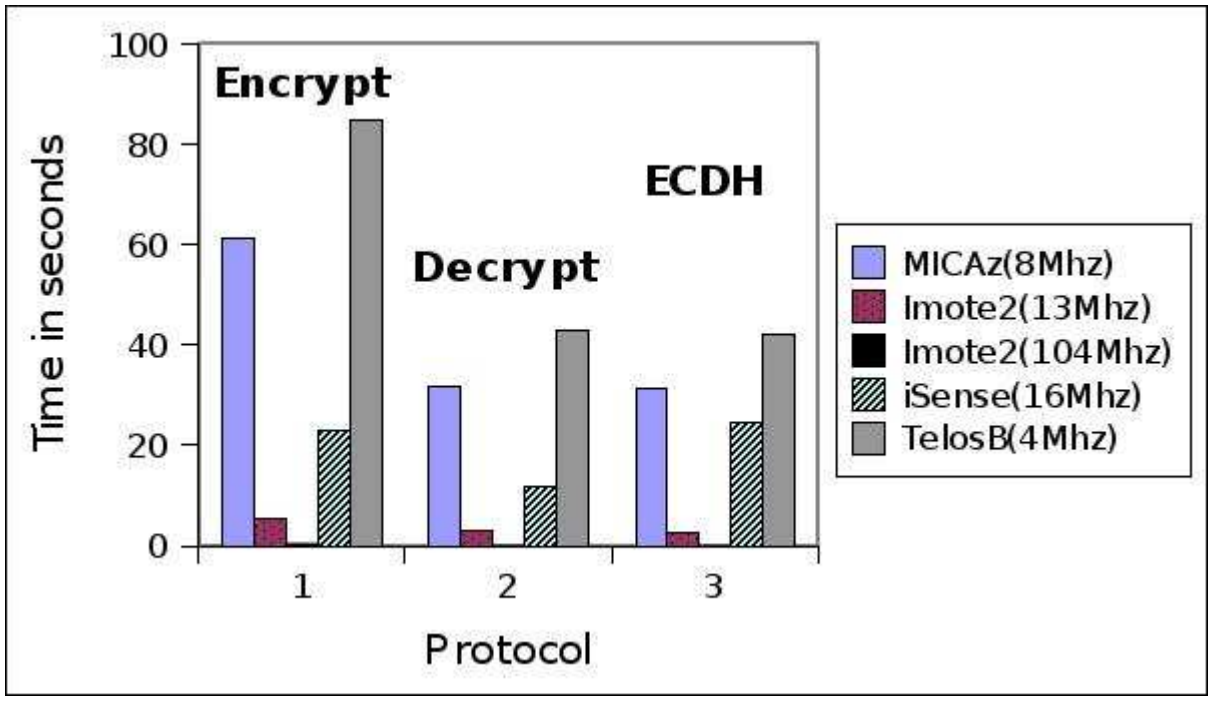



Σχήμα 6.10: Σύγκριση των χρόνων εκτέλεσης των λειτουργιών Encrypt, Decrypt και ECDH ανάμεσα στο πρωτοκολλό μας στη πλατφόρμα iSense και το TinyECC στις διάφορες πλατφόρμες όταν όλες οι τεχνικές βελτιστοποίησης είναι ενεργοποιημένες.

Τέλος, συγκρίναμε τους χρόνους εκτέλεσης των παραπάνω λειτουργιών όταν οι τεχνικές βελτιστοποίησης του TinyECC είναι απενεργοποιημένες.Στην περίπτωση αυτή οι χρόνοι του πρωτοκόλλου μας είναι καλύτεροι σε σχέση με αυτούς της συσκευής MICAz και TelosB.Η συσκευή Imote2 (13Mhz), αν και αργότερη, πετυχαίνει καλύτερους χρόνους διότι η ανάπτυξη κρυπτογραφίας ελλειπτικών καμπυλών γίνεται πάνω από τα πρώτα πεδία $F_p$ και όχι πάνω από τα δυαδικά πεδία $F_{2^p}$ της δικιάς μας υλοποίησης.Η συσκευή Imote2 πετυχαίνει καλύτερους χρόνους καθώς διαθέτει πολύ πιο γρήγορο μικρο-ελεγκτή από αυτόν της συσκευής iSense.Τα αποτελέσματα φαίνονται στην παρακάτω γραφική παράσταση.

### 6.6.3 Συμπεράσματα

Στην εργασία αυτή, παρουσιάσαμε την ανάπτυξη ενός κρυπτογραφικού πρωτοκόλλου με χρήση της κρυπτογράφησης ελλειπτικών καμπυλών (ECC: Elliptic




Σχήμα 6.11: Σύγκριση των χρόνων εκτέλεσης των λειτουργιών Encrypt, Decrypt και ECDH ανάμεσα στο πρωτοκολλό μας στη πλατφόρμα iSense και το TinyECC στις διάφορες πλατφόρμες όταν όλες οι τεχνικές βελτιστοποίησης είναι απενεργοποιημένες.

Curve Cryptography) σε ασύρματα δίκτυα αισθητήρων.Η απόδοση του πρωτοκόλλου αξιολογήθηκε σε πραγματικές συσκευές της σειράς iSense και συγκρίθηκε με αυτήν προηγούμενων εργασιών (Malan, TinyECC).Η απόδοση της εργασίας TinyECC ήταν σε γενικές γραμμές αρκετά καλύτερη λόγω των διαφόρων τεχνικών βελτιστοποίησης που υιοθετήθηκαν κατά την ανάπτυξή της.Ως μελλοντικό στόχο θέτουμε την εφαρμογή αντιστοίχων τεχνικών βελτιστοποίησης για την πλατφόρμα iSense.Ιδιαίτερα, στοχεύουμε στην επίτευξη γρηγορότερου βαθμωτού πολλαπλασιασμού με ελλειπτικές καμπύλες.Όπως είδαμε και νωρίτερα, η πράξη αυτή παίζει σημαντικό ρόλο στη γέννηση δημοσίου κλειδιού, στη διαδικασία συμφωνίας κλειδιού Elliptic Curve Diffie Hellman καθώς στους μηχανισμούς κρυπτογράφησης/αποκρυπτογράφησης δεδομένων.

Από τις παραπάνω μετρήσεις συμπεραίνουμε ότι με το υπάρχον υλικό η ανάπτυξη της κρυπτογραφίας με ελλειπτικές καμπύλες σε ασύρματα δίκτυα αισθητήρων είναι εφικτή.Το βασικό πλεονέκτημα των ελλειπτικών καμπυλών είναι ότι παρέχουν την ίδια ασφάλεια με άλλα συστήματα κρυπτογράφησης δημοσίου

κλειδιού (π.χ. RSA) με χρήση πολύ μικρότερων σε μέγεθος κλειδιών, γεγονός που συνεπάγεται μικρότερη κατανάλωση μνήμης.Παρά το γεγονός αυτό, όπως παρατηρήσαμε και στις παραπάνω μετρήσεις η κρυπτογράφηση με ελλειπτικές καμπύλες καταναλώνει μεγάλο μέρος των πόρων του συστήματος.Ειδικά, οι αλγόριθμοι κρυπτογράφησης/αποκρυπτογράφησης δεδομένων είναι αρκετά χρονοβόροι λόγω του βαθμωτού πολλαπλασιασμού που είναι αρκετά απαιτητική πράξη.Επιπλέον, οι μηχανισμοί κρυπτογράφησης/αποκρυπτογράφησης δεδομένων που αναπτύξαμε αποτελούν κρυπτογραφία δημοσίου κλειδιού.Όπως αναφέραμε και στο **Κεφάλαιο 2**, η συμμετρική κρυπτογραφία συμπληρώνει τη κρυπτογραφία δημοσίου κλειδιού και αντιστρόφως.Έτσι, οι ελλειπτικές καμπύλες μπορούν κάλλιστα να χρησιμοποιηθούν αποδοτικά σε θέματα διανομής και συμφωνίας κλειδιών σε ασύρματα δίκτυα αισθητήρων ενώ για την κρυπτογράφηση των δεδομένων μπορεί να χρησιμοποιηθεί κάποιο συμμετρικό κρυπτοσύστημα το οποίο παρέχει πολύ πιο υψηλούς ρυθμούς κρυπτογράφησης δεδομένων.

# Κεφάλαιο 7

# Πηγαίος Κώδικας

## 7.1 Εισαγωγή

Στο κεφάλαιο αυτό παρουσιάζουμε τον πηγαίο κώδικα του πρωτοκόλλου που αναπτύξαμε καθώς και των συναρτήσεων πάνω στις οποίες βασίζεται.Στο κάθε αρχείο υπάρχουν κατάλληλα σχόλια που επεξηγούν τις λειτουργίες που έχουν υλοποιηθεί.

## 7.2 Το αρχείο "structs.h"

Στο αρχείο αυτό ορίζονται οι κατάλληλες δομές για την κρυπτογράφηση με ελλειπτικές καμπύλες.

```
1  /*************************************************
2   File: "structs.h"
3
4   Description: In this file the appropriate structs
5   for big number and elliptic curve operations are
6   defined
7   *************************************************/
8
9  // constants
10
11 // number of bits in key
12 #define NUMBITS 163
13
14 // maximal number of words required to store a bint;
15 // twice the space necessary to store a key, but allows
16 // for overflow during multiplication, albeit at a cost
17 // (in space and, to some degree, time)
18 #define NUMWORDS (2 * ((int16) ((NUMBITS + 1)/ 8. + 0.5)))
19
```

```
20 // primitives
21
22 // a type primarily for loops' indices
23 typedef int16 index_t;
24
25 // a type for the mote's 8-bit words
26 typedef uint8 word_t;
27
28 // a type for a mote's state
29 typedef uint8 state_t;
30
31 // structures
32
33 // a finite field element from GF(2)[p], modulo some irreducible
34 // polynomial, represented with a polynomial basis as
35 // b_{p-1} x^{p-1} + b_{p-2} x^{p-2} + ... + b_1 x^1 + b_0
36 struct Elt
37 {
38     // b_{p-1} o b_{p-2} o ... o b_1 o b_0;
39     // words are stored as big endian, so b_{p-1} \in val[0] and
40     // b_0 \in val[NUMWORDS-1]
41     uint8 val[NUMWORDS];
42 };
43 typedef struct Elt Elt;
44
45
46 // a curve over GF(2)[p] of the form
47 // y^2 + x y = x^3 + a_4 x^2 + a_6
48 struct Curve
49 {
50     // curve's coefficients
51     Elt a4;
52     Elt a6;
53
54     // modulus for points on the curve
55     uint8 modulus[NUMWORDS];
56
57     // number of bits necessary to represent modulus
58     uint8 bitlength;
59 };
60 typedef struct Curve Curve;
61
62 // a point, (x,y), on a curve
63 struct Point
64 {
65     // point's coordinates
66     Elt x;
67     Elt y;
68 };
```

```
69 typedef struct Point Point;
70
71 // parameters for ECC
72 struct Params
73 {
74     // curve over which ECC will be performed
75     Curve E;
76
77     // a point on E of order r
78     Point G;
79
80     // a positive, prime integer dividing the number of points on
          E
81     uint8 r[NUMWORDS];
82
83     // a positive prime integer, s.t. k = #E/r
84     uint8 k[NUMWORDS];
85
86     // field shall be GF(2)[p]
87     uint16 p;
88
89     // coefficients for irreducible pentanomial,
90     // x^m + x^k3 + x^k2 + x^k1 + 1
91     uint8 pentanomial_k1;
92     uint8 pentanomial_k2;
93     uint8 pentanomial_k3;
94 };
95 typedef struct Params Params;
96
97 // private key for ECC
98 struct PrivKey
99 {
100     // the secret
101     uint8 s[NUMWORDS];
102 };
103 typedef struct PrivKey PrivKey;
104
105 // public key for ECC
106 struct PubKey
107 {
108     // the point
109     Point W;
110 };
111 typedef struct PubKey PubKey;
112
113 // block of data
114 struct DataBlock
115 {
116     // the point
```

```
117     uint8 d[NUMWORDS];
118 };
119 typedef struct DataBlock DataBlock;
```

Listing 7.1: "Το αρχείο structs.h"

## 7.3 Το αρχείο "ecc.h"

Στο αρχείο αυτό υλοποιούνται οι συναρτήσεις για τις πράξεις με μεγάλους ακεραίους, για τις πράξεις με την ελλειπτική καμπύλη (πρόσθεση και πολλαπλασιασμός σημείου), για τις πράξεις πάνω στο πεδίο της ελλειπτικής καμπύλης καθώς και για τις κρυπτογραφικούς μηχανισμούς που αναπτύχθηκαν.

```
1  /*****************************************************
2   * File : "ecc.h"
3   *
4   * Description:
5   * Implementation of big integer routines, point routines,
6   * elliptic curve routines, field routines and crypto
7   * routines
8   *****************************************************/
9
10 #include "structs.h"
11 #include "sha1.h"
12 #include <isense/util/pseudo_random_number_generator.h>
13
14 // hard-coded domain parameters
15 Params params;
16
17 // bint routines
18 inline index_t b_bitlength(uint8 * a);
19 inline void    b_clear(uint8 * a);
20 inline void    b_clearbit(uint8 * a, index_t i);
21 inline int8    b_compareto(uint8 * a, uint8 * b);
22 inline void    b_copy(uint8 * a, uint8 * b);
23 inline bool    b_isequal(uint8 * a, uint8 * b);
24 inline bool    b_iszero(uint8 * a);
25 inline void    b_mod(uint8 * remp, uint8 * modp, int16 lth);
26 inline void    b_setbit(uint8 * a, index_t i);
27 inline void    b_shiftleft(uint8 * a, index_t i, uint8 * b);
28 inline void    b_shiftleft1(uint8 * a, uint8 * b);
29 inline void    b_shiftleft2(uint8 * a, uint8 * b);
30 inline void    b_sub(uint8 *fromp, uint8 *subp, int16 lth);
31 inline bool    b_testbit(uint8 * a, index_t i);
32 inline void    b_xor(uint8 * a, uint8 * b, uint8 * c);
33
34 // point routines
```

```
35 inline void p_clear(Point * P0);
36 inline void p_copy(Point * P0, Point * P1);
37 inline bool p_iszero(Point * P0);
38
39 // curve routines
40 inline void c_add(Point * P0, Point * P1, Point * Q);
41 inline void c_mul(uint8 * n, Point * P0, Point * P1);
42
43 // field routines
44 inline void f_add(uint8 * a, uint8 * b, uint8 * c);
45 inline void f_inv(uint8 * a, uint8 * d);
46 inline void f_mod(uint8 * a, uint8 * b);
47 inline void f_mul(uint8 * a, uint8 * b, uint8 * c);
48
49 // crypto routines
50 inline uint8 * gen_private_key(uint8 * a,uint8 b);
51 inline Point * gen_public_key(uint8 * a,Point * P0);
52 inline Point * gen_shared_secret(uint8 * a,Point * P0, Point * P1
    );
53 inline void eces_encrypt(uint8 * a,uint8 * b,int8 length,Point *
    P0);
54 inline void eces_decrypt(uint8 * a,uint8 * b,int8 length,Point *
    P1);
55 inline void elgamal_encrypt(Point * P0,Point * P1,Point *P2);
56 inline void elgamal_decrypt(Point * P0,Point * P1,Point *P2);
57 inline void psec_encrypt(uint8 * a,uint8 * b,int8 length ,Point *
     P0);
58 inline int8 psec_decrypt(uint8 * a,uint8 * b,int8 length ,Point *
     P0);
59 inline void create_datablock(uint8 * a,int8 length,uint8 b);
60
61 //initialize curve and its parameters
62 inline void init();
63
64 //-----------------------------------------//
65
66 //initialize curve and its parameters
67 inline void init()
68 {
69   // initialize modulus
70   params.p = 163;
71   params.pentanomial_k3 = 7;
72   params.pentanomial_k2 = 6;
73   params.pentanomial_k1 = 3;
74   b_clear(params.E.modulus);
75   b_setbit(params.E.modulus, 163);
76   b_setbit(params.E.modulus, 7);
77   b_setbit(params.E.modulus, 6);
78   b_setbit(params.E.modulus, 3);
```

```
79    b_setbit(params.E.modulus, 0);
80    params.E.bitlength = 164;
81
82    // initialize curve
83    b_clear(params.E.a4.val);
84    params.E.a4.val[NUMWORDS - 1] = (word_t) 0x01;
85    b_clear(params.E.a6.val);
86    params.E.a6.val[NUMWORDS - 1] = (word_t) 0x01;
87
88    // initialize r
89    params.r[NUMWORDS - 21] = (word_t) 0x04;
90    params.r[NUMWORDS - 20] = (word_t) 0x00;
91    params.r[NUMWORDS - 19] = (word_t) 0x00;
92    params.r[NUMWORDS - 18] = (word_t) 0x00;
93    params.r[NUMWORDS - 17] = (word_t) 0x00;
94    params.r[NUMWORDS - 16] = (word_t) 0x00;
95    params.r[NUMWORDS - 15] = (word_t) 0x00;
96    params.r[NUMWORDS - 14] = (word_t) 0x00;
97    params.r[NUMWORDS - 13] = (word_t) 0x00;
98    params.r[NUMWORDS - 12] = (word_t) 0x00;
99    params.r[NUMWORDS - 11] = (word_t) 0x02;
100   params.r[NUMWORDS - 10] = (word_t) 0x01;
101   params.r[NUMWORDS - 9] = (word_t) 0x08;
102   params.r[NUMWORDS - 8] = (word_t) 0xa2;
103   params.r[NUMWORDS - 7] = (word_t) 0xe0;
104   params.r[NUMWORDS - 6] = (word_t) 0xcc;
105   params.r[NUMWORDS - 5] = (word_t) 0x0d;
106   params.r[NUMWORDS - 4] = (word_t) 0x99;
107   params.r[NUMWORDS - 3] = (word_t) 0xf8;
108   params.r[NUMWORDS - 2] = (word_t) 0xa5;
109   params.r[NUMWORDS - 1] = (word_t) 0xef;
110
111   // initialize Gx
112   params.G.x.val[NUMWORDS - 21] = (word_t) 0x02;
113   params.G.x.val[NUMWORDS - 20] = (word_t) 0xfe;
114   params.G.x.val[NUMWORDS - 19] = (word_t) 0x13;
115   params.G.x.val[NUMWORDS - 18] = (word_t) 0xc0;
116   params.G.x.val[NUMWORDS - 17] = (word_t) 0x53;
117   params.G.x.val[NUMWORDS - 16] = (word_t) 0x7b;
118   params.G.x.val[NUMWORDS - 15] = (word_t) 0xbc;
119   params.G.x.val[NUMWORDS - 14] = (word_t) 0x11;
120   params.G.x.val[NUMWORDS - 13] = (word_t) 0xac;
121   params.G.x.val[NUMWORDS - 12] = (word_t) 0xaa;
122   params.G.x.val[NUMWORDS - 11] = (word_t) 0x07;
123   params.G.x.val[NUMWORDS - 10] = (word_t) 0xd7;
124   params.G.x.val[NUMWORDS - 9] = (word_t) 0x93;
125   params.G.x.val[NUMWORDS - 8] = (word_t) 0xde;
126   params.G.x.val[NUMWORDS - 7] = (word_t) 0x4e;
127   params.G.x.val[NUMWORDS - 6] = (word_t) 0x6d;
```

```
128     params.G.x.val[NUMWORDS - 5] = (word_t) 0x5e;
129     params.G.x.val[NUMWORDS - 4] = (word_t) 0x5c;
130     params.G.x.val[NUMWORDS - 3] = (word_t) 0x94;
131     params.G.x.val[NUMWORDS - 2] = (word_t) 0xee;
132     params.G.x.val[NUMWORDS - 1] = (word_t) 0xe8;
133 
134     // initialize Gy
135     params.G.y.val[NUMWORDS - 21] = (word_t) 0x02;
136     params.G.y.val[NUMWORDS - 20] = (word_t) 0x89;
137     params.G.y.val[NUMWORDS - 19] = (word_t) 0x07;
138     params.G.y.val[NUMWORDS - 18] = (word_t) 0x0f;
139     params.G.y.val[NUMWORDS - 17] = (word_t) 0xb0;
140     params.G.y.val[NUMWORDS - 16] = (word_t) 0x5d;
141     params.G.y.val[NUMWORDS - 15] = (word_t) 0x38;
142     params.G.y.val[NUMWORDS - 14] = (word_t) 0xff;
143     params.G.y.val[NUMWORDS - 13] = (word_t) 0x58;
144     params.G.y.val[NUMWORDS - 12] = (word_t) 0x32;
145     params.G.y.val[NUMWORDS - 11] = (word_t) 0x1f;
146     params.G.y.val[NUMWORDS - 10] = (word_t) 0x2e;
147     params.G.y.val[NUMWORDS - 9] = (word_t) 0x80;
148     params.G.y.val[NUMWORDS - 8] = (word_t) 0x05;
149     params.G.y.val[NUMWORDS - 7] = (word_t) 0x36;
150     params.G.y.val[NUMWORDS - 6] = (word_t) 0xd5;
151     params.G.y.val[NUMWORDS - 5] = (word_t) 0x38;
152     params.G.y.val[NUMWORDS - 4] = (word_t) 0xcc;
153     params.G.y.val[NUMWORDS - 3] = (word_t) 0xda;
154     params.G.y.val[NUMWORDS - 2] = (word_t) 0xa3;
155     params.G.y.val[NUMWORDS - 1] = (word_t) 0xd9;
156 
157     // initialize k
158     b_clear(params.k);
159     params.k[NUMWORDS - 1] = (word_t) 0x02;
160 }
161 
162 //-----------------------------------------//
163 // BINT ROUTINES
164 
165 //clear a bint
166 inline void b_clear(uint8 * a)
167 {
168     for (int16 i = 0; i < NUMWORDS; i++)
169     {
170         *(a+i)=0;
171     }
172         //memset(a, 0, NUMWORDS);
173 }
174 
175 //Sets ith bit (where least significant bit is 0th bit) of bint.
176 inline void b_setbit(uint8 * a, index_t i)
```

```
177 {
178     *(a + NUMWORDS - (i / 8) - 1) |= (1 << (i % 8));
179 }
180
181 //Clears ith bit (where least significant bit is 0th bit) of bint
    .
182 inline void b_clearbit(uint8 * a, index_t i)
183 {
184     *(a + NUMWORDS - (i / 8) - 1) &= (0xffff ^ (1 << (i % 8)));
185 }
186
187 //returns true if bint is zero.
188 inline bool b_iszero(uint8 * a)
189 {
190   // index for loop
191   index_t i;
192
193   // determine whether bint is 0; loop ignores top half of a[],
194   // so it'd better be modulo dp.E.modulus already;
195   // casting effectively unrolls loop a bit, saving us some
        cycles
196   for (i = 0 + NUMWORDS/2, a = a + NUMWORDS - 2; i < NUMWORDS; i
        ++, a-=2)
197     if (*((uint16 *) a))
198       return FALSE;
199
200   return TRUE;
201 }
202
203 //a=b
204 inline void b_copy(uint8 * a, uint8 * b)
205 {
206     // index for loop
207     index_t i;
208
209     // copy a[] into b[]; casting effectively unrolls loop a bit,
210     // saving us some cycles
211     for (i = 0; i < NUMWORDS; i += 2)
212         *((uint16 *) (b + i)) = *((uint16 *) (a + i));
213 }
214
215 //c= a XOR b
216 inline void b_xor(uint8 * a, uint8 * b, uint8 * c)
217 {
218   // index for loop
219   index_t i;
220
221   // let c[] = a[] XOR b[]; casting effectively unrolls loop a
        bit,
```

```
222   // saving us some cycles
223   for (i = 0; i < NUMWORDS; i += 2, a += 2, b += 2, c += 2)
224     *((uint16 *) c) = *((uint16 *) a) ^ *((uint16 *) b);
225 }
226
227 //Returns -1 if a < b, 0 if a == b, and 1 if a > b.
228 inline int8 b_compareto(uint8 * a, uint8 * b)
229 {
230
231   // index for loop
232   uint8 lth = NUMWORDS;
233
234   // iterate over a[] and b[], looking for a difference
235   while (lth && *a == *b)
236   {
237     a++;
238     b++;
239     lth--;
240   }
241
242   // if we reached end of a[] and b[], they're the same
243   if (!lth)
244     return 0;
245
246   // if the current byte in a[] is greater than that in b[],
247   // a[] is bigger than b[]
248   else if (*a > *b)
249     return 1;
250
251   // else b[] is bigger than a[]
252   else
253     return -1;
254 }
255
256 //Shifts bint left by n bits, storing result in b.
257 inline void b_shiftleft(uint8 * a, index_t n, uint8 * b)
258 {
259   // index variable
260   index_t i;
261
262   // storage for shift's magnitudes
263   index_t bytes, bits;
264
265   // determine how far to shift whole bytes
266   bytes = n / 8;
267
268   // determine how far to shift bits within bytes or across
269   // pairs of bytes
270   bits = n % 8;
```

```
271
272   // shift whole bytes as appropriate
273   if (bytes > 0)
274   {
275     for (i = bytes; i < NUMWORDS; i++)
276       *(b + i-bytes) = *(a + i);
277     for (i = NUMWORDS - bytes; i < NUMWORDS; i++)
278       *(b + i) = (word_t) 0x00;
279   }
280
281   // else prepare just to shift bits
282   else if (bytes == 0)
283     b_copy(a, b);
284
285   // shift bits as appropriate
286   for (i = 1; i < NUMWORDS; i++)
287     *(b + i - 1) = (*(b + i-1) << bits) | (*(b + i) >> (8 - bits)
          );
288    *(b + NUMWORDS-1) = (*(b + NUMWORDS-1) << bits);
289 }
290
291 //Shifts bint left by 1 bit.  Though a call to this
292 //function is functionally equivalent to one to b_shiftleft(a, 1,
      b),
293 //this version is meant to optimize a common case (shifts by 1).
294 inline void b_shiftleft1(uint8 * a, uint8 * b)
295 {
296   // index variable
297   index_t i;
298
299   if (a != b)
300     b_copy(a, b);
301
302   // shift bits as appropriate; loop is manually unrolled a bit
303   // to save some cycles
304   for (i = 1; i < NUMWORDS - 1; i++)
305   {
306     *(b + i-1) <<= 1;
307     if (*(b + i) & 0x0080)
308       *(b+ i-1) |= 0x0001;
309     i++;
310     *(b + i-1) <<= 1;
311     if (*(b + i) & 0x0080)
312       *(b+ i-1) |= 0x0001;
313   }
314   *(b + NUMWORDS-2) <<= 1;
315   if (*(b + NUMWORDS-1) & 0x0080)
316     *(b + NUMWORDS-2) |= 0x0001;
317   *(b + NUMWORDS-1) <<= 1;
```

```
318 }
319
320 //Shifts bint left by 2 bits.
321 inline void b_shiftleft2(uint8 * a, uint8 * b)
322 {
323   // index variable
324   index_t i;
325
326   if (a != b)
327     b_copy(a, b);
328
329
330   // shift bits as appropriate
331   for (i = 1; i < NUMWORDS; i++)
332   {
333     *(b + i-1) <<= 2;
334     if (*(b + i) & 0x0040)
335       *(b+ i-1) |= 0x0001;
336     if (*(b + i) & 0x0080)
337       *(b+ i-1) |= 0x0002;
338   }
339   *(b + NUMWORDS-1) <<= 2;
340 }
341
342 //Returns the number of bits in the shortest possible
      representation of this bint.
343 inline index_t b_bitlength(uint8 * a)
344 {
345   // index variables;
346   index_t i;
347
348   // local storage
349   uint8 n, x, y;
350
351   // iterate over other bytes, looking for most significant set
       bit;
352   // algorithm from Henry S. Warren Jr., Hacker's Delight
353   for (i = 0; i < NUMWORDS; i++)
354   {
355     x = *(a+i);
356     if (x)
357     {
358       n = 8;
359       y = x >> 4;
360       if (y != 0) {n = n - 4; x = y;}
361       y = x >> 2;
362       if (y != 0) {n = n - 2; x = y;}
363       y = x >> 1;
364       if (y != 0)
```

```
365           return (NUMWORDS - i - 1) * 8 + (8 - (n - 2));
366
367         return (NUMWORDS - i - 1) * 8 + (8 - (n - x));
368       }
369     }
370
371     // if no bits are set, bint is 0
372     return 0;
373 }
374
375 //test if bit i-th bit of bint is set
376 inline bool b_testbit(uint8 * a, index_t i)
377 {
378     return (*(a + NUMWORDS - (i / 8) - 1) & (1 << (i % 8)));
379 }
380
381 //Returns TRUE iff bints are equal.
382 inline bool b_isequal(uint8 * a, uint8 * b)
383 {
384     // index variable
385     index_t i;
386
387     // iterate over bints, looking for a difference
388     for (i = 0; i < NUMWORDS; i++)
389       if (*(a + NUMWORDS - 1 - i) != *(b + NUMWORDS - 1 - i))
390         return FALSE;
391
392     // if no difference found, bints are equal
393     return TRUE;
394 }
395
396 //Subtracts one string of unsigned bytes from another.
397 inline void b_sub(uint8 *fromp, uint8 *subp, int16 lth)
398 {
399     // local variables
400     uint8 *cp, tmp;
401
402     /* step 1 */
403     for (subp += lth - 1, fromp += lth - 1 ; lth--; subp--, fromp
          --)
404     {
405       tmp = *fromp;
406       *fromp -= *subp;
407
408       /* have to borrow */
409       if (*fromp > tmp)
410       {
411         cp = fromp;
412         do
```

```
413         {
414           cp--;
415           (*cp)--;
416         }
417         while(*cp == 0xff);
418       }
419     }
420 }
421
422 //remodularizes by using long division.
423 inline void b_mod(uint8 * remp, uint8 * modp, int16 lth)
424 {
425   uint8 *chremp,     /* ptr to current MSB of remainder */
426       *rtp, *mtp, /* tmp pointers for comparing */
427       *dtp,       /* ptr for dividing */
428   tdiv[2];
429   uint16 tmp, quot;
430   int16 tlth;       /* counter for main loop */
431   uint8 j;
432   uint8 trials[16];
433   uint8 subprod[1 + 96];
434
435   //memset(trials, 0, 16);
436   for(int16 i=0;i<16;i++)
437   {
438     trials[i]=0;
439   }
440
441   modp += NUMWORDS / 2;
442   *trials = *modp >> 1;
443   *(trials + 1) = *modp << 7;
444                   /* step 1 */
445   while (b_compareto(remp, modp) > 0) b_sub(remp, modp, lth);
446                 /* step 2 */
447   for (chremp = remp, tlth = lth; tlth--; chremp++)
448   {
449                 /* step 3 */
450     *tdiv = *chremp;
451     *(tdiv + 1) = *(chremp + 1);
452
453     for (j = 8, dtp = trials, quot = 0; j--; dtp += 2)
454     {
455       quot <<= 1;
456       if (*tdiv > *dtp || (*tdiv == *dtp &&
457         *(tdiv + 1) >= *(dtp + 1)))
458       {
459         b_sub(tdiv, dtp, 2);
460         quot++;
461       }
```

```
462     }
463                     /* step 4 */
464     while (quot > 0xFF) quot = 0xFF;
465     *tdiv = quot - ((quot)? 1: 0);
466                     /* step 5 */
467     if (*tdiv)
468     {
469       //memset(subprod, 0, lth + 1);
470       for(int16 i=0;i<lth+1;i++)
471       {
472         subprod[i]=0;
473       }
474
475       for (mtp = &modp[lth - 1], rtp = &subprod[lth], j = lth; j
              --;
476           mtp--)
477       {
478         tmp = *mtp * *tdiv;
479         tmp += *rtp;
480         *rtp-- = tmp & 0xFF;
481         *rtp = (tmp >> 8);
482       }
483       while (b_compareto(subprod, chremp) > 0)
484       {
485         b_sub(&subprod[1], modp, lth);
486       }
487       b_sub(chremp, subprod, lth + 1);
488     }
489                     /* step 6 */
490     while(*chremp || b_compareto(&chremp[1], modp) > 0)
491       b_sub(&chremp[1], modp, lth);
492   }
493 }
494
495 //-----------------------------------------------------------//
496 //POINT ROUTINES
497
498 //clears a point.
499 inline void p_clear(Point * P0)
500 {
501   // clear each ordinate
502   b_clear(P0->x.val);
503   b_clear(P0->y.val);
504 }
505
506 //Returns TRUE iff P0 == (0,0).
507 inline bool p_iszero(Point * P0)
508 {
509   return (b_iszero(P0->x.val) && b_iszero(P0->y.val));
```

```
510 }
511
512 //P1=P0
513 inline void p_copy(Point * P0, Point * P1)
514 {
515  // copy point's ordinates
516  b_copy(P0->x.val, P1->x.val);
517  b_copy(P0->y.val, P1->y.val);
518 }
519
520 //-------------------------------------------------------//
521 //CURVE ROUTINES
522
523 //Multiplication of P0 by n-> results in P1.
524 //Based on Algorithm IV.1 on p. 63 of
525 //"Elliptic Curves in Cryptography"
526 // by I. F. Blake, G. Seroussi, N. P. Smart.
527 inline void c_mul(uint8 * n, Point * P0, Point * P1)
528 {
529     // index variable
530     index_t i;
531
532     // clear point
533     p_clear(P1);
534
535     // perform multiplication
536     for (i = b_bitlength(n) - 1; i >= 0; i--)
537     {
538         c_add(P1, P1, P1);
539         if (b_testbit(n, i))
540             c_add(P1, P0, P1);
541     }
542 }
543
544 //Q=P1 + P2.
545 //Algorithm 7 in An Overview of Elliptic Curve Cryptography,
      Lopez and Dahab.
546 inline void c_add(Point * P1, Point * P2, Point * Q)
547 {
548   uint8 lambda[NUMWORDS], numerator[NUMWORDS];
549   Point T;
550
551   // 1.  if P1 = 0
552   if (p_iszero(P1))
553   {
554     // Q <-- P2
555     p_copy(P2, Q);
556     return;
557   }
```

```
558
559   // 2.   if P2 = 0
560   if (p_iszero(P2))
561   {
562     // Q <-- P1
563     p_copy(P1, Q);
564     return;
565   }
566
567   // 3.   if x1 = x2
568   if (b_isequal(P1->x.val, P2->x.val))
569   {
570     // if y1 = y2
571     if (b_isequal(P1->y.val, P2->y.val))
572     {
573       // lambda = x1 + y1/x1
574       f_inv(P1->x.val, lambda);
575       f_mul(lambda, P1->y.val, lambda);
576       f_add(lambda, P1->x.val, lambda);
577
578       // x3 = lambda^2 + lambda + a
579       f_mul(lambda, lambda, T.x.val);
580       f_add(T.x.val, lambda, T.x.val);
581       f_add(T.x.val, params.E.a4.val, T.x.val);
582     }
583     else
584     {
585       // Q <-- 0
586       b_clear(T.x.val);
587       b_clear(T.y.val);
588     }
589   }
590   else
591   {
592     // lambda <-- (y2 + y1)/(x2 + x1)
593     f_add(P2->y.val, P1->y.val, numerator);
594     f_add(P2->x.val, P1->x.val, lambda);
595     f_inv(lambda, lambda);
596     f_mul(numerator, lambda, lambda);
597
598     // x3 <-- lambda^2 + lambda + x1 + x2 + a
599     f_mul(lambda, lambda, T.x.val);
600     f_add(T.x.val, lambda, T.x.val);
601     f_add(T.x.val, P1->x.val, T.x.val);
602     f_add(T.x.val, P2->x.val, T.x.val);
603     f_add(T.x.val, params.E.a4.val, T.x.val);
604   }
605
606   // y3 <-- lambda(x1 + x2) + x3 + y1
```

```
607   f_add(P1->x.val, T.x.val, T.y.val);
608   f_mul(T.y.val, lambda, T.y.val);
609   f_add(T.y.val, T.x.val, T.y.val);
610   f_add(T.y.val, P1->y.val, T.y.val);
611
612   // return
613   p_copy(&T, Q);
614 }
615
616 //-------------------------------------------------------------//
617 //FIELD ROUTINES
618
619 // c = a + b.
620 inline void f_add(uint8 * a, uint8 * b, uint8 * c)
621 {
622     b_xor(a, b, c);
623 }
624
625 //c = ab mod f
626 //Algorithm 4 from High-Speed Software Multiplication in F_{2^m}.
627 //a, b, and/or c are allowed to point to the same memory.
628 inline void f_mul(uint8 * a, uint8 * b, uint8 * c)
629 {
630     // local variables
631     index_t i, j, k;
632     uint8 T[NUMWORDS];
633
634     // perform multiplication
635     for (i = 0; i < NUMWORDS; i++)
636         *(T+i) = 0x00;
637     for (j = 7; j >= 0; j--)
638     {
639         for (i = 0; i <= NUMWORDS/2-1; i++)
640             if (b_testbit(a, i*8+j))
641                 for (k = 0; k <= NUMWORDS/2-1; k++)
642                     *(T+(NUMWORDS-1)-(k+i)) ^= *(b+(NUMWORDS-1)-k
                            );
643         if (j != 0) b_shiftleft1(T, T);
644     }
645
646     // modular reduction
647     f_mod(T, c);
648
649 }
650
651
652 // b = a (mod modulus).
653 // a and b are allowed to point to the same memory.
654 // Hardcoded at present with default curve's parameters to save
```

```
    cycles.
655 void f_mod(uint8 * a, uint8 * b)
656 {
657     // local variables
658     index_t blr, shf;
659     int8 comp;
660     uint8 r[NUMWORDS];
661
662     // clear bint
663     b_clear(r);
664
665     // modular reduction
666     comp = b_compareto(a, params.E.modulus);
667     if (comp < 0)
668     {
669         b_copy(a, b);
670         return;
671     }
672     else if (comp == 0)
673     {
674         b_copy(r, b);
675         return;
676     }
677     b_copy(a, r);
678     while ((blr = b_bitlength(r)) >= params.E.bitlength)
679     {
680         shf = blr - params.E.bitlength;
681         *(r + NUMWORDS - ((163+shf) / 8) - 1) ^= (1 << ((163+shf)
                % 8));
682         *(r + NUMWORDS - ((7+shf) / 8) - 1) ^= (1 << ((7+shf) %
                8));
683         *(r + NUMWORDS - ((6+shf) / 8) - 1) ^= (1 << ((6+shf) %
                8));
684         *(r + NUMWORDS - ((3+shf) / 8) - 1) ^= (1 << ((3+shf) %
                8));
685         *(r + NUMWORDS - ((0+shf) / 8) - 1) ^= (1 << ((0+shf) %
                8));
686     }
687     b_copy(r, b);
688 }
689
690
691 // d = a^-1.
692 //Algorithm 8 in "Software Implementation of Elliptic Curve
    Cryptography
693 //Over Binary Fields", D. Hankerson, J.L. Hernandez, A. Menezes.
694 //a and d are allowed to point to the same memory.
695 inline void f_inv(uint8 * a, uint8 * d)
696 {
```

```
697     // local variables
698     index_t i;
699     int8 j;
700     uint8 * ptr;
701     uint8 anonymous[NUMWORDS*5];
702     uint8 * b = anonymous + NUMWORDS;
703     uint8 * c = b + NUMWORDS;
704     uint8 * u = c + NUMWORDS;
705     uint8 * v = u + NUMWORDS;
706
707     // 1.  b <-- 1, c <-- 1, u <-- a, v <-- f
708     for (i = 0; i < NUMWORDS; i++)
709     {
710         *(b+i) = 0x00;
711         *(c+i) = 0x00;
712         *(v+i) = *(params.E.modulus+i);
713     }
714     *(b+NUMWORDS-1) = 0x01;
715     f_mod(a, u);
716
717     // 2.  While deg(u) != 0
718     while (b_bitlength(u) > 1)
719     {
720         // 2.1  j <-- deg(u) - deg(v).
721         j = (b_bitlength(u) - 1) - (b_bitlength(v) - 1);
722
723         // 2.2  If j < 0 then:
724         if (j < 0)
725         {
726             // u <--> v
727             ptr = u;
728             u = v;
729             v = ptr;
730
731             // b <--> c
732             ptr = b;
733             b = c;
734             c = ptr;
735
736             // j <-- -j
737             j = -j;
738         }
739
740         // 2.3  u <-- u + x^jv
741         switch (j)
742         {
743             case 0:
744                 f_add(u, v, u);
745                 f_add(b, c, b);
```

```
746                     break;
747                 case 1:
748                     b_shiftleft1(v, anonymous);
749                     f_add(u, anonymous, u);
750                     b_shiftleft1(c, anonymous);
751                     f_add(b, anonymous, b);
752                     break;
753                 case 2:
754                     b_shiftleft2(v, anonymous);
755                     f_add(u, anonymous, u);
756                     b_shiftleft2(c, anonymous);
757                     f_add(b, anonymous, b);
758                     break;
759                 default:
760                     b_shiftleft(v, j, anonymous);
761                     f_add(u, anonymous, u);
762                     b_shiftleft(c, j, anonymous);
763                     f_add(b, anonymous, b);
764                     break;
765             }
766         }
767         b_copy(b, d);
768 }
769
770 //-------------------------------------------------------------//
771 //CRYPTO ROUTINEs
772
773
774 //generate a sensor nodes private key
775 inline uint8 * gen_private_key(uint8 * a,uint8 b)
776 {
777   isense::PseudoRandomNumberGenerator* R_=new isense::
          PseudoRandomNumberGenerator;
778   for (uint8 i = NUMWORDS/2; i < NUMWORDS; i++)
779   {
780   R_->srand(i^b);
781   *(a+i) = (word_t) R_->rand(255);
782   }
783   //privKeyA.s = privKeyA.s (mod params.r)
784   b_mod(a, params.r, NUMWORDS/2);
785   return a;
786 }
787
788 //generate a sensor nodes public key
789 inline Point * gen_public_key(uint8 * a,Point * P0)
790 {
791   c_mul(a, &params.G, P0);
792   return P0;
793 }
```

```
794
795 //generate a shared secret between two sensor nodes
796 inline Point * gen_shared_secret(uint8 * a,Point * P0, Point * P1
    )
797 {
798   c_mul(a, P0, P1);
799   return P1;
800 }
801
802 //key derivation function
803 void KDF(uint8 *K, int32 K_len, uint8 *Z){
804     int32 len, i;
805     uint8 z[20+4];
806     SHA1Context ctx;
807     uint8 sha1sum[20];
808
809     memcpy(z, Z, 20);
810     memset(z + 20, 0, 3);
811     //KDF
812     len = K_len;
813     i = 1;
814     while(len > 0){
815       z[20 + 3] = i;
816       SHA1Reset(&ctx);
817       SHA1Update(&ctx, z, 20+4);
818       SHA1Digest(&ctx, sha1sum);
819       if(len >= 20){
820   memcpy(K+(i-1)*20, sha1sum, 20);
821       }else{
822   memcpy(K+(i-1)*20, sha1sum, len);
823       }
824       i++;
825       len = len - 20;
826     }
827   }
828
829 //function for mac generation on the data and the key
830 void hmac_sha1(uint8 *text, int32 text_len, uint8 *key, int32
    key_len, uint8 *digest)
831 {
832    SHA1Context context;
833    uint8 k_ipad[65];       /* inner padding -
834                * key XORd with ipad
835                */
836    uint8 k_opad[65];       /* outer padding -
837                * key XORd with opad
838                */
839    int8 i;
840
```

```
841     /*
842      * the HMAC_SHA1 transform looks like:
843      * SHA1(K XOR opad, SHA1(K XOR ipad, text))
844      * where K is an n byte key
845      * ipad is the byte 0x36 repeated 64 times
846      * opad is the byte 0x5c repeated 64 times
847      * and text is the data being protected
848      */
849
850     /* start out by storing key in pads */
851     memcpy(k_ipad, key, key_len);
852     memset(k_ipad + key_len, 0, 65 - key_len);
853     memcpy(k_opad, key, key_len);
854     memset(k_opad + key_len, 0, 65 - key_len);
855
856     /* XOR key with ipad and opad values */
857     for (i=0; i<64; i++) {
858       k_ipad[i] ^= 0x36;
859       k_opad[i] ^= 0x5c;
860     }
861     /*
862      * perform inner SHA1
863      */
864     SHA1Reset(&context);                    /* init context for 1st
           pass */
865     SHA1Update(&context, k_ipad, 64);       /* start with inner pad
           */
866     SHA1Update(&context, text, text_len); /* then text of datagram
           */
867     SHA1Digest(&context, digest);           /* finish up 1st pass
          */
868     /*
869      * perform outer SHA1
870      */
871     SHA1Reset(&context);                    /* init context for 2nd
           pass */
872     SHA1Update(&context, k_opad, 64);      /* start with outer pad
          */
873     SHA1Update(&context, digest, 20);
874     SHA1Digest(&context, digest);           /* then results of 1st
           hash */
875
876  }
877
878 //ecnryption with ECIES scheme
879 inline void psec_encrypt(uint8 * a,uint8 * b,int8 length ,Point *
        P0)
880 {
881   //init variables
```

```
882   uint8 K[80];
883   for(int8 m=0;m<80;m++)
884   {
885     K[m]=0;
886   }
887   PrivKey privKeyA;
888   PubKey pubKeyA;
889   Point secretA;
890   b_clear(privKeyA.s);
891   p_clear(&pubKeyA.W);
892   p_clear(&secretA);
893
894   //generate private key
895   gen_private_key(privKeyA.s,31);
896   //generate public key
897   gen_public_key(privKeyA.s,&pubKeyA.W);
898   //gen shared secret
899   gen_shared_secret(privKeyA.s,P0,&secretA);
900
901   //use KDF to generate K of length message + hash
902   //enckeylen = length, mackeylen = 20
903     //KDF(K, length+20, secretA.x.val);
904  KDF(K, length+20, params.G.x.val);
905   //encrypt the byte block
906     for (int16 i=0; i<length; i++)
907     {
908          b[i] = a[i] ^ K[i];
909     }
910   //b_xor(data, secretA.x.val,data);
911   //hash the encrypted value with SHA-1
912   hmac_sha1(b,length,K+length,20,b+length);
913 }
914
915 //decryption with ECIES scheme
916 inline int8 psec_decrypt(uint8 * a,uint8 * b,int8 length ,Point *
        P0)
917 {
918   //init variables
919   uint8 K[80];
920   for(int8 m=0;m<80;m++)
921   {
922     K[m]=0;
923   }
924   PrivKey privKeyB;
925   Point secretB;
926   b_clear(privKeyB.s);
927   p_clear(&secretB);
928
929   uint8 mac[20];
```

```
930   for(int8 n=0;n<20;n++)
931   {
932     mac[n]=0;
933   }
934   //generate private key
935   gen_private_key(privKeyB.s,29);
936   //gen shared secret
937   gen_shared_secret(privKeyB.s,P0,&secretB);
938
939   //KDF(K,length,secretB.x.val);
940   KDF(K, length+20, params.G.x.val);
941   //hash the encrypted data with the key and check
942   //if they are the same with the hash of the message
943   hmac_sha1(a,length,K+length,20,mac);
944
945   for (int16 i=0; i<20; i++)
946   {
947   if (mac[i] != a[length + i])
948   return 6;
949   }
950   //decrypt
951   for(int16 j=0; j<length; j++)
952   {
953     b[j] = a[j] ^ K[j];
954   }
955   return length;
956 }
957
958 //encrypt a data block with the elliptic curve encryption scheme
959 inline void eces_encrypt(uint8 * a,uint8 * b,int8 length ,Point *
      P0)
960 {
961   PrivKey privKeyA;
962   PubKey pubKeyA;
963   Point secretA;
964   b_clear(privKeyA.s);
965   p_clear(&pubKeyA.W);
966   p_clear(&secretA);
967
968   //generate private key
969   gen_private_key(privKeyA.s,31);
970   //generate public key
971   gen_public_key(privKeyA.s,&pubKeyA.W);
972   //gen shared secret
973   gen_shared_secret(privKeyA.s,P0,&secretA);
974   uint8 data[NUMWORDS];
975   int8 x=-1;
976   //read 20 bytes each time from byte array a
977   for(int16 i=0;i<=length;i=i+NUMWORDS/2)
```

```
978   {
979     x++;
980     b_clear(data);
981     for(int16 k=0;k<NUMWORDS/2-1;k++)
982     {
983       if(i==0)
984         data[NUMWORDS/2+k+1]=*(a+i+k);
985       else
986       data[NUMWORDS/2+k+1]=*(a+i+k-x);
987     }
988     //encrypt the 20byte block
989   f_mul(data, secretA.x.val, data);
990   f_mod(data,data);
991   //return the encrypted block
992     for(int16 j=0;j<NUMWORDS/2;j++)
993     {
994       *(b+i+j)=data[NUMWORDS/2+j];
995     }
996   }
997 }
998
999 //decrypt a data block after encrypted with elliptic curve
      encryption scheme
1000 inline void eces_decrypt(uint8 * a,uint8 * b,int8 length,Point *
      P1)
1001 {
1002   PrivKey privKeyB;
1003   Point secretB;
1004   b_clear(privKeyB.s);
1005   p_clear(&secretB);
1006   //generate private key
1007   gen_private_key(privKeyB.s,29);
1008   //gen shared secret
1009   gen_shared_secret(privKeyB.s,P1,&secretB);
1010
1011   uint8 data[NUMWORDS];
1012   int8 x=-1;
1013   f_inv(secretB.x.val,secretB.x.val);
1014   //read 21 bytes each time from the encrypted byte array a
1015     for(int16 i=0;i<length;i=i+NUMWORDS/2-1)
1016     {
1017       x++;
1018       b_clear(data);
1019         for(int16 k=0;k<NUMWORDS/2;k++)
1020       {
1021         if(i==0)
1022           data[NUMWORDS/2+k]=*(a+i+k);
1023         else
1024         data[NUMWORDS/2+k]=*(a+i+k+x);
```

```
1025        }
1026        //decrypt the 21byte block
1027      f_mul(data, secretB.x.val, data);
1028      f_mod(data,data);
1029      //return the decrypted block
1030        for(int16 j=0;j<NUMWORDS/2-1;j++)
1031        {
1032          *(b+i+j)=data[NUMWORDS/2+j+1];
1033        }
1034      }
1035 }
1036
1037 //user has encoded his message into a point from the finite set
       of points
1038 //in the elliptic curve group
1039 //encrypt a point using elliptic curve el gamal encryption scheme
1040 //by adding the point with the shared secret point
1041 inline void elgamal_encrypt(Point * P0,Point * P1,Point *P2)
1042 {
1043   PrivKey privKeyA;
1044   PubKey pubKeyA;
1045   Point secretA;
1046   b_clear(privKeyA.s);
1047   p_clear(&pubKeyA.W);
1048   p_clear(&secretA);
1049
1050   //generate private key
1051   gen_private_key(privKeyA.s,31);
1052   //generate public key
1053   gen_public_key(privKeyA.s,&pubKeyA.W);
1054   //gen shared secret
1055   gen_shared_secret(privKeyA.s,P1,&secretA);
1056   c_add(P0,&secretA,P2);
1057 }
1058
1059 //decrypt the point after encrypted with elliptic curve el gamal
       scheme
1060 //by adding the point received with the opposite of the shared
       secret point
1061 inline void elgamal_decrypt(Point * P0,Point * P1,Point * P2)
1062 {
1063   PrivKey privKeyB;
1064   Point secretB;
1065   b_clear(privKeyB.s);
1066   p_clear(&secretB);
1067
1068   //generate private key
1069   gen_private_key(privKeyB.s,29);
1070   //gen shared secret
```

```
1071    gen_shared_secret(privKeyB.s,P1,&secretB);
1072
1073    //calucating the opposite of the shared secret point
1074    //in F_2[p] ---> P=(xp,yp) then -P=(xp,xp+yp)
1075    f_add(secretB.x.val,secretB.y.val,secretB.y.val);
1076    c_add(P0,&secretB,P2);
1077 }
1078 //create a byte block for sending
1079 inline void create_datablock(uint8 * a,int8 length,uint8 b)
1080 {
1081    isense::PseudoRandomNumberGenerator* R_=new isense::
           PseudoRandomNumberGenerator;
1082    for (uint8 i = 0; i < length; i++)
1083    {
1084    R_->srand(i^b);
1085    *(a+i) = (word_t) R_->rand(255);
1086    }
1087 }
```

Listing 7.2: "Το αρχείο ecc.h"

## 7.4 Τα αρχεία "sha.h" και "sha1.h"

Στο αρχείο "sha.h" ορίζεται η δομή που αποθηκεύει τις κατάλληλες πληροφορίες για τον αλγόριθμο κατακερματισμού SHA-1, ενώ στο αρχείο "sha1.h" παρουσιάζεται η υλοποίηση του.

```
1  /*
       ************************************************************
2   *  File: "sha.h"
3   *
4   *  Description:
5   *  This is the header file for code which implements the Secure
6   *  Hashing Algorithm 1 as defined in FIPS PUB 180-1 published
7   *  April 17, 1995.
8   *
9   *  Many of the variable names in this code, especially the
10  *  single character names, were used because those were the
        names
11  *  used in the publication.
12  *
13  ************************************************************
        */
14
15 #ifndef _SHA1_H_
16 #define _SHA1_H_
```

```
17
18 #ifndef _SHA_enum_
19 #define _SHA_enum_
20 enum
21 {
22     shaSuccess = 0,
23     shaNull,            /* Null pointer parameter */
24     shaInputTooLong,    /* input data too long */
25     shaStateError       /* called Input after Result */
26 };
27 #endif
28 #define SHA1HashSize 20
29
30 /*
31  *  This structure will hold context information for the SHA-1
32  *  hashing operation
33  */
34 typedef struct SHA1Context
35 {
36     uint32 Intermediate_Hash[SHA1HashSize/4]; /* Message Digest
        */
37
38     uint32 Length_Low;             /* Message length in bits
        */
39     uint32 Length_High;            /* Message length in bits
        */
40
41                                /* Index into message block array
                                   */
42     uint16 Message_Block_Index;
43     uint8 Message_Block[64];       /* 512-bit message blocks
        */
44
45     int8 Computed;                /* Is the digest computed?
                */
46     int8 Corrupted;              /* Is the message digest
        corrupted? */
47 } SHA1Context;
48
49 #endif
```

Listing 7.3: "Το αρχείο sha.h"

```
1 /*********************************************************
2  *  File: "sha1.h"
3  *
4  *  Description:
5  *  This file implements the Secure Hashing Algorithm 1 as
6  *  defined in FIPS PUB 180-1 published April 17, 1995.
```

```
 7  *
 8  *  The SHA-1, produces a 160-bit message digest for a given
 9  *  data stream.  It should take about 2**n steps to find a
10  *  message with the same digest as a given message and
11  *  2**(n/2) to find any two messages with the same digest,
12  *  when n is the digest size in bits.  Therefore, this
13  *  algorithm can serve as a means of providing a
14  *  "fingerprint" for a message.
15  *
16  *  Caveats:
17  *  SHA-1 is designed to work with messages less than 2^64 bits
18  *  long.  Although SHA-1 allows a message digest to be generated
19  *  for messages of any number of bits less than 2^64, this
20  *  implementation only works with messages with a length that is
21  *  a multiple of the size of an 8-bit character.
22  ************************************************************
        */
23
24 #include "sha.h"
25 #define METHOD2
26 /*
27  *  Define the circular shift macro
28  */
29 #define SHA1CircularShift(bits,word) \
30                 ((((word) << (bits)) & 0xFFFFFFFF) | \
31                 ((word) >> (32-(bits))))
32
33 //function prototypes
34 void SHA1Reset(SHA1Context *);
35 int8 SHA1Digest( SHA1Context *context, uint8 Message_Digest[
       SHA1HashSize]);
36 int8 SHA1Update(SHA1Context *context, const uint8 *message_array,
        uint32 length);
37 void SHA1ProcessMessageBlock(SHA1Context *);
38 void SHA1PadMessage(SHA1Context *);
39
40 /*
41  *  SHA1Reset
42  *
43  *  Description:
44  *      This function will initialize the SHA1Context in
       preparation
45  *      for computing a new message digest.
46  *
47  *  Parameters:
48  *      context: [in/out]
49  *          The context to reset.
50  *
51  *  Returns:
```

```
52  *       Nothing.
53  *
54  */
55 void SHA1Reset(SHA1Context *context)
56 {
57     context->Length_Low             = 0;
58     context->Length_High            = 0;
59     context->Message_Block_Index    = 0;
60
61     context->Intermediate_Hash[0]       = 0x67452301;
62     context->Intermediate_Hash[1]       = 0xEFCDAB89;
63     context->Intermediate_Hash[2]       = 0x98BADCFE;
64     context->Intermediate_Hash[3]       = 0x10325476;
65     context->Intermediate_Hash[4]       = 0xC3D2E1F0;
66
67     context->Computed   = 0;
68     context->Corrupted  = 0;
69 }
70
71 /*
72  *  SHA1Digest
73  *
74  *  Description:
75  *      This function will return the 160-bit message digest into
        the
76  *      Intermediate_Hash array within the SHA1Context provided
77  *
78  *  Parameters:
79  *      context: [in/out]
80  *          The context to use to calculate the SHA-1 hash.
81  *
82  *  Returns:
83  *      1 if successful, 0 if it failed.
84  *
85  *  Comments:
86  *
87  */
88 int8 SHA1Digest( SHA1Context *context, uint8 Message_Digest[
      SHA1HashSize])
89   {
90     uint8 i;
91
92     if (!context || !Message_Digest){
93       return shaNull;
94     }
95
96     if (context->Corrupted){
97       return context->Corrupted;
98     }
```

```
99
100     if (!context->Computed){
101       SHA1PadMessage(context);
102       for(i=0; i<64; ++i){
103   /* message may be sensitive, clear it out */
104   context->Message_Block[i] = 0;
105       }
106       context->Length_Low = 0;     /* and clear length */
107       context->Length_High = 0;
108       context->Computed = 1;
109
110     }
111
112     for(i = 0; i < SHA1HashSize; ++i){
113       Message_Digest[i] = context->Intermediate_Hash[i>>2]
114   >> 8 * ( 3 - ( i & 0x03 ) );
115     }
116
117     return 1;
118   }
119
120 /*
121  *  SHA1Update
122  *
123  *  Description:
124  *      This function accepts an array of octets as the next
       portion of
125  *      the message.
126  *
127  *  Parameters:
128  *      context: [in/out]
129  *          The SHA-1 context to update
130  *      message_array: [in]
131  *          An array of characters representing the next portion
       of the
132  *          message.
133  *      length: [in]
134  *          The length of the message in message_array
135  *
136  *  Returns:
137  *      Nothing.
138  *
139  *  Comments:
140  *
141  */
142 int8 SHA1Update(SHA1Context *context, const uint8 *message_array,
       uint32 length)
143 {
144     if (!length){
```

```
145       return shaSuccess;
146     }
147 
148     if (!context || !message_array){
149       return shaNull;
150     }
151 
152     if (context->Computed){
153       context->Corrupted = shaStateError;
154 
155       return shaStateError;
156     }
157 
158     if (context->Corrupted){
159       return context->Corrupted;
160     }
161     while(length-- && !context->Corrupted){
162       context->Message_Block[context->Message_Block_Index++] =
163   (*message_array & 0xFF);
164 
165       context->Length_Low += 8;
166       if (context->Length_Low == 0){
167         context->Length_High++;
168         if (context->Length_High == 0){
169     /* Message is too long */
170     context->Corrupted = 1;
171         }
172       }
173 
174       if (context->Message_Block_Index == 64){
175         SHA1ProcessMessageBlock(context);
176       }
177 
178       message_array++;
179     }
180 
181     return shaSuccess;
182   }
183 
184 /*
185  *  SHA1ProcessMessageBlock
186  *
187  *  Description:
188  *      This function will process the next 512 bits of the
     message
189  *      stored in the Message_Block array.
190  *
191  *  Parameters:
192  *      None.
```

```
193  *
194  *  Returns:
195  *      Nothing.
196  *
197  *  Comments:
198  *      Many of the variable names in the SHAContext, especially
         the
199  *      single character names, were used because those were the
         names
200  *      used in the publication.
201  *
202  *
203  */
204
205   void SHA1ProcessMessageBlock(SHA1Context *context)
206   {
207     const uint32 K[] =    {       /* Constants defined in SHA-1
            */
208       0x5A827999,
209       0x6ED9EBA1,
210       0x8F1BBCDC,
211       0xCA62C1D6
212     };
213     uint8        t;                  /* Loop counter
                        */
214     uint32       temp;               /* Temporary word value
                    */
215#ifdef METHOD2
216     uint8        s;
217     uint32       W[16];
218#else
219     uint32       W[80];              /* Word sequence
                           */
220#endif
221     uint32       A, B, C, D, E;      /* Word buffers
                           */
222
223     /*
224      *  Initialize the first 16 words in the array W
225      */
226     for(t = 0; t < 16; t++){
227       W[t] = ((uint32)context->Message_Block[t * 4]) << 24;
228       W[t] |= ((uint32)context->Message_Block[t * 4 + 1]) << 16;
229       W[t] |= ((uint32)context->Message_Block[t * 4 + 2]) << 8;
230       W[t] |= ((uint32)context->Message_Block[t * 4 + 3]);
231     }
232
233#ifndef METHOD2
234     for(t = 16; t < 80; t++){
```

```
235         W[t] = SHA1CircularShift(1,W[t-3] ^ W[t-8] ^ W[t-14] ^ W[t
                -16]);
236       }
237 #endif
238
239       A = context->Intermediate_Hash[0];
240       B = context->Intermediate_Hash[1];
241       C = context->Intermediate_Hash[2];
242       D = context->Intermediate_Hash[3];
243       E = context->Intermediate_Hash[4];
244
245       for(t = 0; t < 20; t++){
246 #ifdef METHOD2
247         s = t & 0x0f;
248         if (t >= 16){
249   W[s] = SHA1CircularShift(1,
250           W[(s + 13) & 0x0f] ^ \
251           W[(s + 8) & 0x0f] ^ \
252           W[(s + 2) & 0x0f] ^ \
253           W[s]);
254         }
255         temp =  SHA1CircularShift(5,A) +
256    ((B & C) | ((~B) & D)) + E + W[s] + K[0];
257 #else
258         temp =  SHA1CircularShift(5,A) +
259    ((B & C) | ((~B) & D)) + E + W[t] + K[0];
260 #endif
261         E = D;
262         D = C;
263         C = SHA1CircularShift(30,B);
264
265         B = A;
266         A = temp;
267       }
268
269       for(t = 20; t < 40; t++){
270 #ifdef METHOD2
271         s = t & 0x0f;
272         W[s] = SHA1CircularShift(1,
273                 W[(s + 13) & 0x0f] ^   \
274                 W[(s + 8) & 0x0f] ^    \
275                 W[(s + 2) & 0x0f] ^    \
276                 W[s]);
277         temp = SHA1CircularShift(5,A) + (B ^ C ^ D) + E + W[s] + K
                [1];
278 #else
279         temp = SHA1CircularShift(5,A) + (B ^ C ^ D) + E + W[t] + K
                [1];
280 #endif
```

```
281         E = D;
282         D = C;
283         C = SHA1CircularShift(30,B);
284         B = A;
285         A = temp;
286       }
287
288       for(t = 40; t < 60; t++){
289 #ifdef METHOD2
290         s = t & 0x0f;
291         W[s] = SHA1CircularShift(1,
292                 W[(s + 13) & 0x0f] ^   \
293                 W[(s + 8) & 0x0f] ^    \
294                 W[(s + 2) & 0x0f] ^    \
295                 W[s]);
296         temp = SHA1CircularShift(5,A) +
297     ((B & C) | (B & D) | (C & D)) + E + W[s] + K[2];
298 #else
299         temp = SHA1CircularShift(5,A) +
300     ((B & C) | (B & D) | (C & D)) + E + W[t] + K[2];
301 #endif
302         E = D;
303         D = C;
304         C = SHA1CircularShift(30,B);
305         B = A;
306         A = temp;
307       }
308
309       for(t = 60; t < 80; t++){
310 #ifdef METHOD2
311         s = t & 0x0f;
312         W[s] = SHA1CircularShift(1,
313                 W[(s + 13) & 0x0f] ^   \
314                 W[(s + 8) & 0x0f] ^    \
315                 W[(s + 2) & 0x0f] ^    \
316                 W[s]);
317         temp = SHA1CircularShift(5,A) + (B ^ C ^ D) + E + W[s] + K
              [3];
318 #else
319         temp = SHA1CircularShift(5,A) + (B ^ C ^ D) + E + W[t] + K
              [3];
320 #endif
321         E = D;
322         D = C;
323         C = SHA1CircularShift(30,B);
324         B = A;
325         A = temp;
326       }
327
```

```
328     context->Intermediate_Hash[0] += A;
329     context->Intermediate_Hash[1] += B;
330     context->Intermediate_Hash[2] += C;
331     context->Intermediate_Hash[3] += D;
332     context->Intermediate_Hash[4] += E;
333
334     context->Message_Block_Index = 0;
335   }
336
337 /*
338  *  SHA1PadMessage
339  *
340  *  Description:
341  *      According to the standard, the message must be padded to
        an even
342  *      512 bits.  The first padding bit must be a '1'.  The last
         64
343  *      bits represent the length of the original message.  All
        bits in
344  *      between should be 0.  This function will pad the message
345  *      according to those rules by filling the Message_Block
        array
346  *      accordingly.  It will also call SHA1ProcessMessageBlock()
347  *      appropriately.  When it returns, it can be assumed that
        the
348  *      message digest has been computed.
349  *
350  *  Parameters:
351  *      context: [in/out]
352  *          The context to pad
353  *
354  *  Returns:
355  *      Nothing.
356  *
357  *  Comments:
358  *
359  */
360
361 void SHA1PadMessage(SHA1Context *context)
362   {
363     /*
364      *  Check to see if the current message block is too small to
          hold
365      *  the initial padding bits and length.  If so, we will pad
          the
366      *  block, process it, and then continue padding into a
          second
367      *  block.
368      */
```

```
369     if (context->Message_Block_Index > 55){
370       context->Message_Block[context->Message_Block_Index++] = 0
              x80;
371       while(context->Message_Block_Index < 64){
372   context->Message_Block[context->Message_Block_Index++] = 0;
373       }
374
375       SHA1ProcessMessageBlock(context);
376
377       while(context->Message_Block_Index < 56){
378   context->Message_Block[context->Message_Block_Index++] = 0;
379       }
380     }else{
381       context->Message_Block[context->Message_Block_Index++] = 0
              x80;
382       while(context->Message_Block_Index < 56){
383   context->Message_Block[context->Message_Block_Index++] = 0;
384       }
385     }
386
387     /*
388      *  Store the message length as the last 8 octets
389      */
390     context->Message_Block[56] = context->Length_High >> 24;
391     context->Message_Block[57] = context->Length_High >> 16;
392     context->Message_Block[58] = context->Length_High >> 8;
393     context->Message_Block[59] = context->Length_High;
394     context->Message_Block[60] = context->Length_Low >> 24;
395     context->Message_Block[61] = context->Length_Low >> 16;
396     context->Message_Block[62] = context->Length_Low >> 8;
397     context->Message_Block[63] = context->Length_Low;
398
399     SHA1ProcessMessageBlock(context);
400   }
```

Listing 7.4: "Το αρχείο sha1.h"

## 7.5 Το αρχείο "TimeProviderDemoApplication.cpp"

Το αρχείο αυτό περιέχει την εφαρμογή iSense που εκτελεί το πρωτόκολλό μας.Όπως αναφέραμε στο προηγούμενο κεφάλαιο, δύο συσκευές της σειράς iSense αφού δημιουργήσουν το ιδιωτικό και δημόσιο κλειδί τους εκτελούν τη διαδικασία συμφωνίας κλειδιού Diffie-Hellman.Έτσι καταλήγουν σε ένα νέο κλειδί με το οποίο ανταλλάσουν κρυπτογραφημένα μπλοκ δεδομένων.

```
/*************************************************
 File: "TimeProviderDemoApplication.cpp"

 Description:
 1. 2 nodes create their private and public keys
 2. Elliptic curve diffie hellman : they create a shared
    secret (point on the curve)
 3. they exchange encrypted data blocks and decrypt
 *************************************************/

#include <isense/application.h>
#include <isense/config.h>
#include <isense/os.h>
#include <isense/dispatcher.h>
#include <isense/radio.h>
#include <isense/task.h>
#include <isense/timeout_handler.h>
#include <isense/isense.h>
#include <isense/uart.h>
#include <isense/dispatcher.h>
#include <isense/time.h>
#include <isense/button_handler.h>
#include <isense/sleep_handler.h>
#include <isense/modules/core_module/core_module.h>
#include <isense/util/util.h>
#include <isense/protocols/routing/flooding.h>
#include <isense/protocols/time_sync/confirm_time.h>
#include "ecc.h"

#define MILLISECONDS 4000
#define PUB_KEY_MSG 150
#define BLOCK_ENCR_MSG 151
#define TASK_GENKEYS 1
#define TASK_SEND_PUBLIC 2
#define TASK_GEN_SHAREDKEY 3
#define TASK_SEND_ENCRYPTED 4
//-----------------------------------------
/**
 */
using namespace isense;

class TimeProviderDemoApplication :
  public isense::Application,
  public isense::Receiver,
  public isense::Sender,
  public isense::TimeoutHandler,
  public isense::Task

```

```
49 {
50 public:
51   TimeProviderDemoApplication(isense::Os& os);
52
53   virtual ~TimeProviderDemoApplication() ;
54
55   ///From isense::Application
56   virtual void boot (void) ;
57
58   ///From isense::Receiver
59   virtual void receive (uint8 len, const uint8 * buf, uint16
         src_addr, uint16 dest_addr, uint16 lqi, uint8 seq_no, uint8
          interface) ;
60
61   ///From isense::Sender
62   virtual void confirm (uint8 state, uint8 tries, isense::Time
         time) ;
63
64   ///From isense::Task
65   virtual void execute( void* userdata ) ;
66
67   ///From isense::TimeoutHandler
68   virtual void timeout( void* userdata ) ;
69 private:
70   TimeSync* ts_;
71
72   // Alice's private key
73   PrivKey privKeyA;
74
75   // Alice's public key
76   PubKey pubKeyA;
77
78   //Alice sends
79   DataBlock sendvalue;
80
81   //bob receives
82   DataBlock encryptedvalue;
83   DataBlock receivedvalue;
84   DataBlock decryptedvalue;
85
86   Point Recover;
87   // Alice and Bob's shared secret
88   Point secretA;
89   Point ElGamalSend;
90   Point ElGamalReceive;
91 };
92
93 //--------------------------------------------------------
94   TimeProviderDemoApplication::
```

```
95    TimeProviderDemoApplication(isense::Os& os)
96    : isense::Application(os)
97    {
98    }
99
100 //--------------------------------------------------------------
101   TimeProviderDemoApplication::
102   ~TimeProviderDemoApplication()
103
104   {
105   }
106
107 //--------------------------------------------------------------
108 void
109   TimeProviderDemoApplication::
110   boot(void)
111   {
112         os_.debug("I am device %d and i Boot",os().id());
113         os_.dispatcher().add_receiver(this);
114         os_.allow_sleep(false);
115
116         // initialize storage for keys
117       p_clear(&pubKeyA.W);
118       b_clear(privKeyA.s);
119       b_clear(sendvalue.d);
120       b_clear(receivedvalue.d);
121       b_clear(encryptedvalue.d);
122       b_clear(decryptedvalue.d);
123
124       // initialize storage for secret
125       p_clear(&secretA);
126       p_clear(&Recover);
127       p_clear(&ElGamalSend);
128       p_clear(&ElGamalReceive);
129
130       //initialize curve and its parameters
131       init();
132       os().debug("Sensor preloaded key (elliptic curve base point
              ) is: ");
133       os().debug("X coefficient : %d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%
              d%d%d%d%d%d%d",
134       params.G.x.val[NUMWORDS/2],params.G.x.val[NUMWORDS/2+1],
              params.G.x.val[NUMWORDS/2+2],params.G.x.val[NUMWORDS
              /2+3],
135       params.G.x.val[NUMWORDS/2+4],params.G.x.val[NUMWORDS/2+5],
              params.G.x.val[NUMWORDS/2+6],params.G.x.val[NUMWORDS
              /2+7],
136       params.G.x.val[NUMWORDS/2+8],params.G.x.val[NUMWORDS/2+9],
              params.G.x.val[NUMWORDS/2+10],params.G.x.val[NUMWORDS
```

```
            /2+11],
137         params.G.x.val[NUMWORDS/2+12],params.G.x.val[NUMWORDS
                /2+13],params.G.x.val[NUMWORDS/2+14],params.G.x.val[
                NUMWORDS/2+15],
138         params.G.x.val[NUMWORDS/2+16],params.G.x.val[NUMWORDS
                /2+17],params.G.x.val[NUMWORDS/2+18],params.G.x.val[
                NUMWORDS/2+19],
139         params.G.x.val[NUMWORDS/2+20]);
140
141         os().debug("Y coefficient : %d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%
                d%d%d%d%d%d%d",
142         params.G.y.val[NUMWORDS/2],params.G.y.val[NUMWORDS/2+1],
                params.G.y.val[NUMWORDS/2+2],params.G.y.val[NUMWORDS
                /2+3],
143         params.G.y.val[NUMWORDS/2+4],params.G.y.val[NUMWORDS/2+5],
                params.G.y.val[NUMWORDS/2+6],params.G.y.val[NUMWORDS
                /2+7],
144         params.G.y.val[NUMWORDS/2+8],params.G.y.val[NUMWORDS/2+9],
                params.G.y.val[NUMWORDS/2+10],params.G.y.val[NUMWORDS
                /2+11],
145         params.G.y.val[NUMWORDS/2+12],params.G.y.val[NUMWORDS
                /2+13],params.G.y.val[NUMWORDS/2+14],params.G.y.val[
                NUMWORDS/2+15],
146         params.G.y.val[NUMWORDS/2+16],params.G.y.val[NUMWORDS
                /2+17],params.G.y.val[NUMWORDS/2+18],params.G.y.val[
                NUMWORDS/2+19],
147         params.G.y.val[NUMWORDS/2+20]);
148
149         ts_=new TimeSync(os());
150         //caling the task for private and public key generation
151           os_.add_task(this, (void*)TASK_GENKEYS);
152     }
153
154 //-----------------------------------------------
155 void
156     TimeProviderDemoApplication::
157     execute( void* userdata )
158     {
159     ts_->launch_time_sync();
160
161     //task for node private and public key generation
162     if ((uint32)userdata == TASK_GENKEYS)
163     {
164     os().debug("Generating private and public key!");
165     //generating private key
166     gen_private_key(privKeyA.s,os().id());
167
168     os().debug("My private key is: ");
169     os().debug("%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d",
```

```
170     privKeyA.s[NUMWORDS/2],privKeyA.s[NUMWORDS/2+1],privKeyA.s[
            NUMWORDS/2+2],privKeyA.s[NUMWORDS/2+3],
171     privKeyA.s[NUMWORDS/2+4],privKeyA.s[NUMWORDS/2+5],privKeyA.s[
            NUMWORDS/2+6],privKeyA.s[NUMWORDS/2+7],
172     privKeyA.s[NUMWORDS/2+8],privKeyA.s[NUMWORDS/2+9],privKeyA.s[
            NUMWORDS/2+10],privKeyA.s[NUMWORDS/2+11],
173     privKeyA.s[NUMWORDS/2+12],privKeyA.s[NUMWORDS/2+13],privKeyA.s[
            NUMWORDS/2+14],privKeyA.s[NUMWORDS/2+15],
174     privKeyA.s[NUMWORDS/2+16],privKeyA.s[NUMWORDS/2+17],privKeyA.s[
            NUMWORDS/2+18],privKeyA.s[NUMWORDS/2+19],
175     privKeyA.s[NUMWORDS/2+20]);
176
177     //generating public key
178     gen_public_key(privKeyA.s,&pubKeyA.W);
179
180     os().debug("My public key is: ");
181     os().debug("X coefficient : %d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%
            d%d%d%d",
182     pubKeyA.W.x.val[NUMWORDS/2],pubKeyA.W.x.val[NUMWORDS/2+1],
            pubKeyA.W.x.val[NUMWORDS/2+2],pubKeyA.W.x.val[NUMWORDS
            /2+3],
183     pubKeyA.W.x.val[NUMWORDS/2+4],pubKeyA.W.x.val[NUMWORDS/2+5],
            pubKeyA.W.x.val[NUMWORDS/2+6],pubKeyA.W.x.val[NUMWORDS
            /2+7],
184     pubKeyA.W.x.val[NUMWORDS/2+8],pubKeyA.W.x.val[NUMWORDS/2+9],
            pubKeyA.W.x.val[NUMWORDS/2+10],pubKeyA.W.x.val[NUMWORDS
            /2+11],
185     pubKeyA.W.x.val[NUMWORDS/2+12],pubKeyA.W.x.val[NUMWORDS/2+13],
            pubKeyA.W.x.val[NUMWORDS/2+14],pubKeyA.W.x.val[NUMWORDS
            /2+15],
186     pubKeyA.W.x.val[NUMWORDS/2+16],pubKeyA.W.x.val[NUMWORDS/2+17],
            pubKeyA.W.x.val[NUMWORDS/2+18],pubKeyA.W.x.val[NUMWORDS
            /2+19],
187     pubKeyA.W.x.val[NUMWORDS/2+20]);
188
189     os().debug("Y coefficient : %d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%
            d%d%d%d",
190     pubKeyA.W.y.val[NUMWORDS/2],pubKeyA.W.y.val[NUMWORDS/2+1],
            pubKeyA.W.y.val[NUMWORDS/2+2],pubKeyA.W.y.val[NUMWORDS
            /2+3],
191     pubKeyA.W.y.val[NUMWORDS/2+4],pubKeyA.W.y.val[NUMWORDS/2+5],
            pubKeyA.W.y.val[NUMWORDS/2+6],pubKeyA.W.y.val[NUMWORDS
            /2+7],
192     pubKeyA.W.y.val[NUMWORDS/2+8],pubKeyA.W.y.val[NUMWORDS/2+9],
            pubKeyA.W.y.val[NUMWORDS/2+10],pubKeyA.W.y.val[NUMWORDS
            /2+11],
193     pubKeyA.W.y.val[NUMWORDS/2+12],pubKeyA.W.y.val[NUMWORDS/2+13],
            pubKeyA.W.y.val[NUMWORDS/2+14],pubKeyA.W.y.val[NUMWORDS
            /2+15],
```

```
194   pubKeyA.W.y.val[NUMWORDS/2+16],pubKeyA.W.y.val[NUMWORDS/2+17],
          pubKeyA.W.y.val[NUMWORDS/2+18],pubKeyA.W.y.val[NUMWORDS
          /2+19],
195   pubKeyA.W.y.val[NUMWORDS/2+20]);
196
197   //calling the task for sending the public key to neighbor
          devices
198   os_.add_task(this,(void*)TASK_SEND_PUBLIC);
199   }
200
201   //task for sending the public key
202   if ((uint32)userdata == TASK_SEND_PUBLIC)
203   {
204   os().debug("Sending the public key with elgamal encryption!");
205   //encrypting the point first
206   elgamal_encrypt(&pubKeyA.W,&params.G,&ElGamalSend);
207
208   //preparing the buffer about to send
209   uint8 outbuffer[43];
210   outbuffer[0]=PUB_KEY_MSG;
211   for(int16 i=1;i<22;i++)
212   {
213   outbuffer[i]=ElGamalSend.x.val[NUMWORDS/2+i-1];
214   }
215   for(int16 i=22;i<43;i++)
216   {
217   outbuffer[i]=ElGamalSend.y.val[NUMWORDS/2+i-22];
218   }
219   //sending the message with the encrypted public key through the
          radio
220   os().radio().send(ISENSE_RADIO_BROADCAST_ADDR,43,outbuffer,
          Radio::ISENSE_RADIO_HEADER_OPTION_NONE,NULL);
221   }
222
223   //task fore shared secret generation
224   if((uint32)userdata == TASK_GEN_SHAREDKEY)
225   {
226   os().debug("Generating Shared Secret!");
227   gen_shared_secret(privKeyA.s,&ElGamalReceive,&secretA);
228
229   os().debug("The shared secret key is: ");
230   os().debug("X coefficient : %d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%
          d%d%d%d",
231   secretA.x.val[NUMWORDS/2],secretA.x.val[NUMWORDS/2+1],secretA.x
          .val[NUMWORDS/2+2],secretA.x.val[NUMWORDS/2+3],
232   secretA.x.val[NUMWORDS/2+4],secretA.x.val[NUMWORDS/2+5],secretA
          .x.val[NUMWORDS/2+6],secretA.x.val[NUMWORDS/2+7],
233   secretA.x.val[NUMWORDS/2+8],secretA.x.val[NUMWORDS/2+9],secretA
          .x.val[NUMWORDS/2+10],secretA.x.val[NUMWORDS/2+11],
```

```
234     secretA.x.val[NUMWORDS/2+12],secretA.x.val[NUMWORDS/2+13],
            secretA.x.val[NUMWORDS/2+14],secretA.x.val[NUMWORDS/2+15],
235     secretA.x.val[NUMWORDS/2+16],secretA.x.val[NUMWORDS/2+17],
            secretA.x.val[NUMWORDS/2+18],secretA.x.val[NUMWORDS/2+19],
236     secretA.x.val[NUMWORDS/2+20]);
237
238     os().debug("Y coefficient : %d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%
            d%d%d%d",
239     secretA.y.val[NUMWORDS/2],secretA.y.val[NUMWORDS/2+1],secretA.y
            .val[NUMWORDS/2+2],secretA.y.val[NUMWORDS/2+3],
240     secretA.y.val[NUMWORDS/2+4],secretA.y.val[NUMWORDS/2+5],secretA
            .y.val[NUMWORDS/2+6],secretA.y.val[NUMWORDS/2+7],
241     secretA.y.val[NUMWORDS/2+8],secretA.y.val[NUMWORDS/2+9],secretA
            .y.val[NUMWORDS/2+10],secretA.y.val[NUMWORDS/2+11],
242     secretA.y.val[NUMWORDS/2+12],secretA.y.val[NUMWORDS/2+13],
            secretA.y.val[NUMWORDS/2+14],secretA.y.val[NUMWORDS/2+15],
243     secretA.y.val[NUMWORDS/2+16],secretA.y.val[NUMWORDS/2+17],
            secretA.y.val[NUMWORDS/2+18],secretA.y.val[NUMWORDS/2+19],
244     secretA.y.val[NUMWORDS/2+20]);
245
246     //call the task for sending an encrypted data block
247     os().add_task_in(Time(MILLISECONDS),this,(void*)
            TASK_SEND_ENCRYPTED);
248     //os_.add_task(this, (void*)TASK_SEND_ENCRYPTED);
249     }
250
251     //task for encrypting a data block and sending it
252     if ((uint32)userdata == TASK_SEND_ENCRYPTED)
253     {
254     os().debug("Creating a data block!");
255     b_clear(sendvalue.d);
256
257     //create a data block of 21 bytes
258     create_datablock(sendvalue.d,20,os().id());
259     os().debug("Original Sending Value is:");
260     os().debug("%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d",
261     sendvalue.d[0],sendvalue.d[1],sendvalue.d[2],sendvalue.d[3],
262     sendvalue.d[4],sendvalue.d[5],sendvalue.d[6],sendvalue.d[7],
263     sendvalue.d[8],sendvalue.d[9],sendvalue.d[10],sendvalue.d[11],
264     sendvalue.d[12],sendvalue.d[13],sendvalue.d[14],sendvalue.d
            [15],
265     sendvalue.d[16],sendvalue.d[17],sendvalue.d[18],sendvalue.d
            [19]);
266
267     b_clear(encryptedvalue.d);
268     //encrypt the data block
269     eces_encrypt(sendvalue.d,encryptedvalue.d,20,&secretA);
270     //psec_encrypt(sendvalue.d,encryptedvalue.d,20,&secretA);
271
```

```
272    os().debug("Encrypted Sending Value is:");
273    os().debug("%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d",
274    encryptedvalue.d[NUMWORDS/2],encryptedvalue.d[NUMWORDS/2+1],
           encryptedvalue.d[NUMWORDS/2+2],encryptedvalue.d[NUMWORDS
           /2+3],
275    encryptedvalue.d[NUMWORDS/2+4],encryptedvalue.d[NUMWORDS/2+5],
           encryptedvalue.d[NUMWORDS/2+6],encryptedvalue.d[NUMWORDS
           /2+7],
276    encryptedvalue.d[NUMWORDS/2+8],encryptedvalue.d[NUMWORDS/2+9],
           encryptedvalue.d[NUMWORDS/2+10],encryptedvalue.d[NUMWORDS
           /2+11],
277    encryptedvalue.d[NUMWORDS/2+12],encryptedvalue.d[NUMWORDS
           /2+13],encryptedvalue.d[NUMWORDS/2+14],encryptedvalue.d[
           NUMWORDS/2+15],
278    encryptedvalue.d[NUMWORDS/2+16],encryptedvalue.d[NUMWORDS
           /2+17],encryptedvalue.d[NUMWORDS/2+18],encryptedvalue.d[
           NUMWORDS/2+19],
279    encryptedvalue.d[NUMWORDS/2+20]);
280
281    //preparing the buffer to send the encrypted value
282    uint8 buffer[22];
283    buffer[0]=BLOCK_ENCR_MSG;
284    for(int16 i=1;i<22;i++)
285    {
286    buffer[i]=encryptedvalue.d[NUMWORDS/2+i-1];
287    }
288
289    //sending the encrypted data block through the radio
290    os().radio().send(ISENSE_RADIO_BROADCAST_ADDR,22,buffer,Radio::
           ISENSE_RADIO_HEADER_OPTION_NONE,NULL);
291    }
292    }
293
294
295 //-------------------------------------------------------
296 void
297    TimeProviderDemoApplication::
298    receive (uint8 len, const uint8 * buf, uint16 src_addr, uint16
           dest_addr, uint16 lqi, uint8 seq_no, uint8 interface)
299    {
300    if(buf[0]==PUB_KEY_MSG) //if it is a message containing a
           public key
301      {
302      os_.debug("i received %d bytes from node with id %x.",len,
             src_addr);
303
304      //recovering the encrypted public key
305      for(int16 i=1;i<22;i++)
306      {
```

```
307         Recover.x.val[NUMWORDS/2+i-1]=buf[i];
308       }
309       for(int16 i=22;i<43;i++)
310       {
311         Recover.y.val[NUMWORDS/2+i-22]=buf[i];
312       }
313
314       //decrypting the point with elgamal decryption
315       elgamal_decrypt(&Recover,&params.G,&ElGamalReceive);
316
317       //calling the task for generating shared secret
318       os_.add_task(this,(void*)TASK_GEN_SHAREDKEY);
319       }
320
321     if(buf[0]==BLOCK_ENCR_MSG) //if it is a message with encrypted
          data block
322       {
323       os_.debug("i received %d bytes from node with id %x.",len,
            src_addr);
324
325       //recovering the encrypted data
326       for(int16 i=1;i<22;i++)
327       {
328         receivedvalue.d[NUMWORDS/2+i-1]=buf[i];
329       }
330
331       os().debug("Received value is : ");
332       os().debug("%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d",
333       receivedvalue.d[NUMWORDS/2],receivedvalue.d[NUMWORDS/2+1],
            receivedvalue.d[NUMWORDS/2+2],receivedvalue.d[NUMWORDS
            /2+3],
334       receivedvalue.d[NUMWORDS/2+4],receivedvalue.d[NUMWORDS/2+5],
            receivedvalue.d[NUMWORDS/2+6],receivedvalue.d[NUMWORDS
            /2+7],
335       receivedvalue.d[NUMWORDS/2+8],receivedvalue.d[NUMWORDS/2+9],
            receivedvalue.d[NUMWORDS/2+10],receivedvalue.d[NUMWORDS
            /2+11],
336       receivedvalue.d[NUMWORDS/2+12],receivedvalue.d[NUMWORDS
            /2+13],receivedvalue.d[NUMWORDS/2+14],receivedvalue.d[
            NUMWORDS/2+15],
337       receivedvalue.d[NUMWORDS/2+16],receivedvalue.d[NUMWORDS
            /2+17],receivedvalue.d[NUMWORDS/2+18],receivedvalue.d[
            NUMWORDS/2+19],
338       receivedvalue.d[NUMWORDS/2+20]);
339
340       //decrypting the data block
341       eces_decrypt(receivedvalue.d,decryptedvalue.d,20,&secretA);
342       //psec_decrypt(receivedvalue.d,decryptedvalue.d,20,&secretA);
343
```

```
344       os().debug("Decrypted value is : ");
345       os().debug("%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d",
346       decryptedvalue.d[NUMWORDS/2],decryptedvalue.d[NUMWORDS/2+1],
              decryptedvalue.d[NUMWORDS/2+2],decryptedvalue.d[NUMWORDS
              /2+3],
347       decryptedvalue.d[NUMWORDS/2+4],decryptedvalue.d[NUMWORDS
              /2+5],decryptedvalue.d[NUMWORDS/2+6],decryptedvalue.d[
              NUMWORDS/2+7],
348       decryptedvalue.d[NUMWORDS/2+8],decryptedvalue.d[NUMWORDS
              /2+9],decryptedvalue.d[NUMWORDS/2+10],decryptedvalue.d[
              NUMWORDS/2+11],
349       decryptedvalue.d[NUMWORDS/2+12],decryptedvalue.d[NUMWORDS
              /2+13],decryptedvalue.d[NUMWORDS/2+14],decryptedvalue.d[
              NUMWORDS/2+15],
350       decryptedvalue.d[NUMWORDS/2+16],decryptedvalue.d[NUMWORDS
              /2+17],decryptedvalue.d[NUMWORDS/2+18],decryptedvalue.d[
              NUMWORDS/2+19],
351       decryptedvalue.d[NUMWORDS/2+20]);
352       }
353
354     }
355
356 //-----------------------------------------------------------------
357 void
358     TimeProviderDemoApplication::
359     confirm (uint8 state, uint8 tries, isense::Time time)
360     {
361
362     }
363
364 //-----------------------------------------------------------------
365 void
366     TimeProviderDemoApplication::
367     timeout( void* userdata )
368     {
369     os_.add_task(this,(void*)TASK_SEND_PUBLIC);
370     //os_.add_timeout_in(Time(MILLISECONDS),this,NULL);
371     }
372
373 //-----------------------------------------------------------------
374
375 isense::Application* application_factory(isense::Os& os)
376 {
377     return new TimeProviderDemoApplication(os);
378 }
379
380 //-----------------------------------------------------------------
```

Listing 7.5: "Το αρχείο TimeProviderDemoApplication.cpp"

# Βιβλιογραφία


[1] 'Handbook of Applied Cryptography', by Alfred J. Menezes, Paul C. van Oorschot and Scott A. Vanstone, October 1996

[2] 'Elliptic Curves in Cryptography' by I. Blake, G. Seroussi and N. Smart, London Mathematical Society Lecture Note Series 265, Cambrigde University Press, 1999

[3] 'An Overview of Elliptic Curve Cryptography' by Julio López and Ricardo Dahab, 2000

[4] Tutorial on Elliptic Curves : http://www.certicom.com/index.php/10-introduction

[5] Θεωρία και εφαρμογές κρυπτογραφικών συστημάτων δημοσίου κλειδιού βασισμένων σε ελλειπτικές καμπύλες, Διδακτορική διατριβή της Ε.Κωνσταντίνου, Πάτρα, Ιούνιος 2005

[6] 'The advantage of Elliptic Curve Cryptography for wireless security' by K. Lauter - Wireless Communications, IEEE

[7] 'A Method for Obtaining Digital Signatures and Public-Key Cryptosystems' by Rivest, R.A. Shamir and L. Adleman, 1978, Communications of the ACM 21 (2): 120–126

[8] 'A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms' by Taher ElGamal, IEEE Transactions on Information Theory, v. IT-31, n. 4, 1985, pp469–472 or CRYPTO 84, pp10–18, Springer-Verlag

[9] 'New Directions in Cryptography' by W. Diffie and M. E. Hellman, IEEE Transactions on Information Theory, vol. IT-22, Nov. 1976, pp: 644-654

[10] 'SPINS: security protocols for sensor networks' by Adrian Perrig, Robert Szewczyk, Victor Wen, David Culler and J. D. Tygar, Proceedings of the 7th annual international conference on Mobile computing and networking, p.189-199, July 2001, Rome, Italy

[11] 'LEAP: efficient security mechanisms for large-scale distributed sensor networks' by Sencun Zhu, Sanjeev Setia and Sushil Jajodia, Proceedings of the 10th ACM conference on Computer and communications security, 2003, Washington D.C., USA

[12] 'TinySec: A Link Layer Security Architecture for Wireless Sensor Networks' by C. Karlof, N. Sastry, and D. Wagner, Second ACM Conference on Embedded Networked Sensor Systems, 2004, Baltimore, Maryland

[13] 'Secure Routing in Sensor Networks: Attacks and Countermeasures' by C. Karlof and D. Wagner, Proceedings of First IEEE Workshop on Sensor Network Protocols and Applications, May 2003

[14] 'A key-management scheme for distributed sensor networks' by Laurent Eschenauer and Virgil D. Gligor, Proceedings of the 9th ACM conference on Computer and communications security, November 18-22, 2002, Washington, DC, USA

[15] 'Shawn: The fast, highly customizable sensor network simulator' by Fekete, S.P. Kroller, A. Fischer, S. Pfisterer, Braunschweig University of Technology, Networked Sensing Systems, 2007. INSS '07. Fourth International Conference on Volume Issue, 6-8 June 2007 Page(s): 299 - 299

[16] Shawn Introduction and Tutorial : http://shawn.sourceforge.net/

[17] 'iSense: Modular hardware and software for wireless sensor networks' by Carsten Buschmann, Coalesenses GmbH

[18] Coalesenses Gmbh: http://www.coalesenses.com/

[19] 'A Software Library for Elliptic Curve Cryptography' by E. Konstantinou, Y. Stamatiou, and C. Zaroliagis in Algorithms - ESA 2002 (Engineering and Applications Track), Lecture Notes in Computer Science 2461 (Springer-Verlag, 2002), pp.625-637

[20] 'The GNU MP Bignum Library' : http://gmplib.org/

[21] 'A Public-Key Infrastructure for Key Distribution in TinyOS Based on Elliptic Curve Cryptography' by David J. Malan, Matt Welsh, and Michael D. Smith. First IEEE International Conference on Sensor and Ad Hoc Communications and Networks, Santa Clara, California, October 2004.

[22] 'TinyECC: A Configurable Library for Elliptic Curve Cryptography in Wireless Sensor Networks' by An Liu and Peng Ning, in Proceedings of the 7th International Conference on Information Processing in Sensor Networks (IPSN 2008), SPOTS Track, pages 245–256, April 2008

[23] IEEE P1363: Asymmetric Encryption : http://grouper.ieee.org/groups/1363/P1363a/Encryption.html