ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΑΛΕΞΑΝΔΡΕΙΟ ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΘΕΣΣΑΛΟΝΙΚΗΣ

ΜΕΘΟΔΟΙ ΚΡΥΠΤΟΓΡΑΦΙΑΣ ΣΤΗΝ ΠΛΗΡΟΦΟΡΙΚΗ

Πτυχιακή Εργασία του

Σκόδρα Θωμά

Επιβλέπων καθηγητής: Διαμαντάρας Κωνσταντίνος

Τμήμα Πληροφορικής
Α.Τ.Ε.Ι.Θ.
Θεσσαλονίκη

Φεβρουάριος 2009

Ευχαριστίες

Στο σημείο αυτό θα ήθελα να ευχαριστήσω τον καθηγητή κύριο Κωνσταντίνο Διαμαντάρα, επιβλέποντα της πτυχιακής μου εργασίας, που μου έδωσε την ευκαιρία να ασχοληθώ με το συγκεκριμένο θέμα και με καθοδήγησε με τις συμβουλές του στη τελική διεκπεραίωση αυτού. Επίσης, θα ήθελα να ευχαριστήσω τους γονείς μου Ιωάννη και Αγγελική για την ηθική υποστήριξη που με προσέφεραν κατά τη διάρκεια της εργασίας και όλων αυτών των ετών που σπούδασα στη σχολή. Τέλος, αξίζει να μνημονευθεί η υποστήριξη που μου παρείχαν η αδερφή μου Βαΐα και όλοι οι καθηγητές της σχολής που τυγχάνει να είναι από τους καλύτερους στην Ελλάδα.

ΠΕΡΙΛΗΨΗ

Η κρυπτογραφία ως λέξη παράγεται από τα συνθετικά κρυπτός, που σημαίνει μυστικός και γράφω. Η τέχνη της κρυπτογραφίας μέσα στο χρόνο μεταμορφώθηκε σε επιστήμη, η οποία συνδέεται άμεσα με την πληροφορική. Όλες, ή σχεδόν όλες, οι εφαρμογές της πληροφορικής περιέχουν πληροφορίες που πρέπει να αποκρυφτούν από κάποιο τρίτο πρόσωπο. Η κρυπτογραφία στη πληροφορική υλοποιείται με διάφορες τεχνικές που αποσκοπούν τόσο στη καλύτερη ασφάλεια και προστασία των δεδομένων, όσο και στην κατανάλωση λιγότερης υπολογιστικής ισχύος αλλά και στο μικρότερο κόστος υλοποίησης των αλγορίθμων που χρησιμοποιούνται για το σκοπό αυτό. Στη παρούσα εργασία γίνεται περιγραφή και ανάλυση των σπουδαιότερων τεχνικών κρυπτογράφησης στη πληροφορική. Η αρχή γίνεται με την ανάλυση των σημαντικότερων αλγόριθμων συμμετρικής κρυπτογράφησης, όπως ο αλγόριθμος του Καίσαρα, ο DES, ο IDEA και ο AES. Στη συνέχεια παρουσιάζεται ο αλγόριθμος ανταλλαγής κλειδιών του Diffie-Hellman, ο αλγόριθμος δημοσίου κλειδιού RSA, αναλύεται η λειτουργία των hash συναρτήσεων και περιγράφεται ο αλγόριθμος ασύμμετρης κρυπτογράφησης του ElGamal, ο οποίος χρησιμοποιεί το πρόβλημα του διακριτού λογαρίθμου. Η εργασία ολοκληρώνεται με την ανάλυση της θεωρίας και της χρήσης των ελλειπτικών καμπυλών στην κρυπτογραφία. Οι ελλειπτικές καμπύλες αποτελούν τη βάση για τη δημιουργία μιας νέας κατηγορίας κρυπτογραφικών συστημάτων δημοσίου κλειδιού. Τα τελευταία χρόνια, όλο και περισσότερες κρυπτογραφικές εφαρμογές βασίζονται στη χρήση των ελλειπτικών καμπυλών. Η ανάλυση και παρουσίαση της θεωρίας των ελλειπτικών καμπυλών βασίστηκε στην ιστοσελίδα της εταιρείας Certicom, η οποία ασχολείται με την κρυπτογραφία και τη χρήση των ελλειπτικών καμπυλών σε αυτή. Στην παρούσα εργασία, αφού έγινε εκτεταμένη παρουσίαση της θεωρίας των αριθμών, της θεωρίας των ελλειπτικών καμπυλών και του προβλήματος του διακριτού λογαρίθμου στις ελλειπτικές καμπύλες, αναλύθηκαν ορισμένες από τις σπουδαιότερες εφαρμογές των ελλειπτικών καμπυλών στη κρυπτογραφία. Συγκεκριμένα παρουσιάστηκαν ο αλγόριθμος δημιουργίας ψηφιακής

υπογραφής με τη χρήση ελλειπτικών καμπυλών (ECDSA), ο αλγόριθμος ανταλλαγής κλειδιού με τη χρήση ελλειπτικών καμπυλών των Diffie - Hellman (ECDH) και ο ολοκληρωμένος αλγόριθμος κρυπτογράφησης – αποκρυπτογράφησης με τη χρήση ελλειπτικών καμπυλών (ECIES). Αρκετές υλοποιήσεις έχουν έρθει στο προσκήνιο και πολλοί επιστήμονες της πληροφορικής ασχολούνται αποκλειστικά με αυτόν τον τομέα της κρυπτογραφίας. Μετά την ανάλυση των αλγορίθμων και την βιβλιογραφική έρευνα που κάναμε κρίνουμε ότι οι ελλειπτικές καμπύλες θα μπορούσαν και πρέπει να χρησιμοποιηθούν για τη δημιουργία διαφόρων κρυπτογραφικών εφαρμογών στην πληροφορική.

Summary

Cryptography (or cryptology; from Greek «κρυπτός», kryptos, "hidden, secret"; and γράφω, gráphō, "I write", or -λογία, -logia, respectively) is the practice and study of hiding information. The art of cryptography in time was transformed to a science which has straight connection with information science. All, or almost all, applications in informatics contain information which has to be concealed from a third party person. Cryptography in informatics is implemented with several techniques which aim either at better security and privacy of data either at less consumption of computing power and cost of implementation of the algorithms that are used for this purpose. In this thesis, we demonstrate and analyze the foremost cryptographic techniques in informatics. In the beginning we analyze the most important algorithms of symmetric cryptography, as Caesar's algorithm, DES, IDEA and AES. Afterwards, we develop Diffie-Hellman's key exchange protocol, public key RSA algorithm, hash function's operation and finally we describe how asymmetric ElGamal's algorithm works using Discrete Algorithm Problem. This thesis is accomplished with the elliptic curves theory and use in cryptography. Elliptic curves constitute the fundamental for a formation of a new category of public key cryptographic systems. Last years, more and more cryptographic applications depend on the use of elliptic curves. The analysis and the presentation of the theoretical part of this thesis base on the web site of Certicom, which work on cryptography and the elliptic curves' use on it. In this thesis, after we had developed the number theory, the elliptic curves theory and the discrete logarithm problem in elliptic curves, we analyzed some of the most useful applications of elliptic curves in cryptography. Specifically, we showed the algorithm of digital signature generation ECDSA (Elliptic Curve Digital Signature Algorithm), the algorithm of encryption – decryption ECIES (Elliptic Curve Integrated Encryption Scheme) and the algorithm of key agreement ECDH (Elliptic Curve Diffie – Hellman). A lot of implementations came out and several computer scientists work exclusively with this section of cryptography.

After algorithms' analysis and bibliographic review we conclude that elliptic curves should be used for the creation of several cryptographic applications in information science.

ΠΕΡΙΕΧΟΜΕΝΑ

ΚΕΦΑΛΑΙΟ 1

1. Εισαγωγή

1.1 Ιστορική Αναδρομή.

1.1.1 Η Κρυπτογραφία μέσα στο χρόνο.

Η κρυπτογραφία ξεκίνησε πριν από περίπου 3500 χρόνια ως μια τέχνη απόκρυψης μυστικών που στηριζόταν στην επινοητικότητα του ανθρώπου που επιθυμούσε να διαφυλάξει τα μυστικά του.

Παρακάτω παρουσιάζονται συνοπτικά κάποια γεγονότα που αποδεικνύουν την εφαρμογή της κρυπτογραφίας σε διάφορες πτυχές της ζωής και της ιστορίας του ανθρώπου [1]:

- 2500 π.Χ.: Τα Ιερογλυφικά στις πυραμίδες της Αιγύπτου αποτελούσαν κρυπτογραφημένα κείμενα.
- 500 π.Χ.: Οι Σπαρτιάτες κάνανε τατουάζ στο σώμα τους κωδικοποιημένα μηνύματα για να μη τα καταλαβαίνει ο αντίπαλος.
- 150 π.Χ.: Στο Ρωμαϊκό στρατό εφαρμόστηκε ο αλγόριθμος του Καίσαρα για την κρυπτογράφηση μηνυμάτων.
- 1000 μ.Χ.: Εμφανίστηκαν μονοαλφαβητικές μέθοδοι μέσα στο Κοράνι.
- 1465 μ.Χ., Ιταλία: Εμφάνιση κρυπτογραφημένων μηνυμάτων για επικοινωνία μεταξύ πόλεων από τον Leon Alberti.
- 1900 μ.Χ., Αγγλία: Δημιουργία κρυπτογραφημένων μηνυμάτων της Βασίλισσας Σκωτίας Μαρίας για μυστική επικοινωνία με το στρατό της.
- Α’ Παγκόσμιος πόλεμος: Γερμανία – Δημιουργία του συστήματος Enigma.
- Β’ Παγκόσμιος πόλεμος: Ιαπωνία – Δημιουργία του συστήματος Purple.
- Β’ Παγκόσμιος πόλεμος: ΗΠΑ – Δημιουργία του συστήματος Sigaba.
- Β’ Παγκόσμιος πόλεμος: Αγγλία – Δημιουργία του συστήματος TypeX.

1.1.2 Η Κρυπτογραφία σήμερα.

Σήμερα το αντικείμενο της κρυπτογραφίας δεν περιορίζεται απλά και μόνο στην απόκρυψη μηνυμάτων. Η σύγχρονη κρυπτογραφία, ωθούμενη και από την ραγδαία ανάπτυξη της περίφημης «κοινωνίας των πληροφοριών», ασχολείται πια με τη μεθοδολογική ανάλυση και εκτίμηση οποιουδήποτε συστήματος χρησιμοποιείται για την επικοινωνία και την οικονομική, εργασιακή και κοινωνική συνδιαλλαγή μεταξύ ανθρώπων και οργανισμών. Και όχι μόνο αυτό. Η μοντέρνα κρυπτογραφία, είναι πια σε θέση να παρέχει εγγυήσεις και εκτιμήσεις όσον αφορά την ασφάλεια της διαρκώς αυξανόμενης γκάμας μεθόδων και πρωτοκόλλων ασφαλούς επίτευξης κάθε μορφής επικοινωνίας. Μέσα από την αλληλεπίδρασή της με τις επιστήμες της πληροφορικής και των μαθηματικών *η μοντέρνα κρυπτογραφία έχει μεταμορφωθεί από μία τέχνη σε μία επιστήμη*, βασισμένη σε αυστηρές προσεγγίσεις και μεθόδους ανάλυσης [2].

1.2 Κρυπτογραφία και Κρυπτοσυστήματα.

1.2.1 Τι είναι η Κρυπτογραφία.

Στη διεθνή βιβλιογραφία προτείνονται διάφοροι ορισμοί της κρυπτογραφίας [3]. Ο πιο διαδεδομένος αναφέρεται στο πρόβλημα της μυστικής επικοινωνίας:

Έτσι, *η* ***κρυπτογραφία*** *μπορούμε να πούμε ότι μελετά τρόπους με τους οποίους μπορούμε να μετασχηματίσουμε ένα μήνυμα σε φαινομενικά ακατάληπτη μορφή.* Ο ορισμός αυτός αν και μπορεί να καλύψει στο μέγιστο βαθμό τη χρήση της κρυπτογραφίας από την εποχή της αρχαίας Αιγύπτου μέχρι και τη Βιομηχανική Επανάσταση, στην εποχή της Πληροφορικής έχει βασικές ελλείψεις. Ο ορισμός που δόθηκε από τον Rivest (1990) εισάγει την έννοια του *αντιπάλου* και είναι ίσως ο πιο ακριβής και πλήρης ορισμός.

Από το σημείο αυτό, με τον όρο κρυπτογραφία θα εννοούμε το εξής:

Η κρυπτογραφία ασχολείται με την επικοινωνία παρουσία αντιπάλων. Η ύπαρξη αντιπάλου σε κάποια επικοινωνία είναι η βασική αιτία ύπαρξης και εφαρμογής της κρυπτογραφίας. Εκτός από την επιθυμία μας να κρύψουμε ένα μήνυμα από τα μάτια των αντιπάλων, θα πρέπει με κάποιο τρόπο να μην αλλοιωθεί το μήνυμα μας, ή αν αλλοιωθεί να γίνει αντιληπτό από τον παραλήπτη, και επίσης να φτάσει στον πραγματικό παραλήπτη και όχι σε κάποιον που τον υποδύεται.

1.2.2 Τι είναι το Κρυπτοσύστημα.

Η αρχική μορφή ενός μηνύματος, αποτελεί το **απλό κείμενο** (plaintext), ενώ το κρυπτογραφημένο κείμενο αποτελεί το **κρυπτοκείμενο** (ciphertext). Ο μετασχηματισμός του απλού κειμένου σε κρυπτοκείμενο ονομάζεται **κρυπτογράφηση** (encryption) ενώ ο μετασχηματισμός του κρυπτοκειμένου σε απλό κείμενο, αντίστροφα ονομάζεται **αποκρυπτογράφηση** (decryption). Οι διαδικασίες κρυπτογράφησης και αποκρυπτογράφησης υλοποιούνται με τον **αλγόριθμο κρυπτογράφησης** και **αποκρυπτογράφησης** αντίστοιχα. Οι δύο αυτοί αλγόριθμοι συνιστούν τον **κρυπταλγόριθμο** (cipher). Η διαδικασία κρυπτογράφησης (και αποκρυπτογράφησης) απαιτεί μια επιπλέον ποσότητα πληροφορίας που την ονομάζουμε **κλειδί** (key). Η ύπαρξη του κλειδιού αποτελεί και τη διαφορά της κρυπτογράφησης με την **κωδικοποίηση** (encoding).

Αναλυτικότερα, η κρυπτογράφηση και η αποκρυπτογράφηση ενός κειμένου μπορεί να πραγματοποιηθεί με επιτυχία μόνον από τον κάτοχο του σωστού κλειδιού. Ο όρος *κλειδί* είναι πολύ εύστοχος καθότι το κλειδί παραπέμπει σε κάτι μυστικό, που έχει συγκεκριμένους κατόχους, και είναι αναγκαίο για να κλειδώνει και να ξεκλειδώνει κλειδαριές. Έτσι, ένας αλγόριθμος κρυπτογράφησης μπορεί να παρομοιαστεί με μια κλειδαριά, η οποία χρησιμοποιείται για να φυλάξει ένα μήνυμα. Όποιος έχει το κλειδί μπορεί χωρίς μεγάλη προσπάθεια να ανοίξει την κλειδαριά και να διαβάσει το μήνυμα. *Η περιγραφή των διαδικασιών κρυπτογράφησης και αποκρυπτογράφησης αποτελούν το* ***κρυπτοσύστημα***.

Ο αντίπαλος ενός κρυπτοσυστήματος θα επικεντρωθεί στο να ανακαλύψει το σωστό κλειδί, δηλαδή το κλειδί εκείνο το οποίο θα μπορέσει να ανοίξει την κλειδαριά και να διαβάσει το μήνυμα.

Τέλος, **κρυπτανάλυση** είναι η επιστήμη που ασχολείται με την αποκρυπτογράφηση του κρυπτοκειμένου χωρίς την κατοχή του κλειδιού. Ο αντίπαλος μπορεί να μη γνωρίζει το κλειδί, το οποίο μπορεί να είναι πολύ καλά θαμμένο μέσα στο σύστημα, αλλά μπορεί να εκμεταλλευτεί την πρόσβαση του σε αυτό και να επιτύχει την αποκρυπτογράφηση με άλλο τρόπο. Συνήθως όμως, ο αντίπαλος ενδιαφέρεται να ανακαλύψει το κλειδί για να μπορεί να αποκρυπτογραφήσει όλα τα μηνύματα που ενδεχομένως στάλθηκαν με τη χρήση του κλειδιού αυτού [3].

1.3 Υπηρεσίες της Κρυπτογραφίας.

1.3.1 Τι διασφαλίζει η Κρυπτογραφία.

Κατά την αποστολή μηνυμάτων μεταξύ δύο ενδιαφερόμενων μέσα από ένα μη ασφαλές κανάλι πρέπει να διασφαλιστούν οι αρχές της **Εμπιστευτικότητας** (**Confidentiality**), της **Αυθεντικοποίησης** (**Authentication**), της **Ακεραιότητας** (**Integrity**) και της **μη αποποίηση ευθύνης** (**Non-Repudiation**) [4].

- Με τον όρο **εμπιστευτικότητα** εννοούμε την προστασία των δεδομένων ενάντια σε μη εξουσιοδοτημένη πρόσβαση ή γνωστοποίηση τους. Η εμπιστευτικότητα μπορεί να παρομοιασθεί με έναν αδιαφανή φάκελο.
- Η **αυθεντικοποίηση** είναι η επιβεβαίωση της ταυτότητας ενός ατόμου ή η επιβεβαίωση της πηγής αποστολής των πληροφοριών. Η υπηρεσία αυτή παρέχεται από μηχανισμούς κρυπτογραφίας όπως είναι οι ψηφιακές υπογραφές [5].
- Η **ακεραιότητα** είναι η προστασία των δεδομένων ενάντια σε μη εξουσιοδοτημένη τροποποίηση ή αντικατάσταση τους.

- Τέλος, η **μη αποποίηση ευθύνης** (Non-Repudiation) συνδυάζει τις υπηρεσίες της αυθεντικότητας και της ακεραιότητας που παρέχονται σε μια τρίτη οντότητα. Έτσι, εξασφαλίζεται ότι τα μέρη που εμπλέκονται σε μια ηλεκτρονική επικοινωνία δεν θα μπορούν να αρνηθούν εκ των υστέρων τη συμμετοχή τους σε αυτήν [6].

1.3.2 Εφαρμογές της Κρυπτογραφίας στη πληροφορική.

Με τη βοήθεια των υπηρεσιών, που προσφέρει η κρυπτογραφία, μπορούμε να έχουμε διάφορες ασφαλείς εφαρμογές στη πληροφορική όπως:

- Ασφαλές Ηλεκτρονικό Ταχυδρομείο.
- Πρόσβαση σε ασφαλείς δικτυακούς τόπους.
- Προστασία ευαίσθητων δεδομένων σε γραμματείες τμημάτων και διοικητικούς φορείς.
- Προστασία ερευνητικών δεδομένων.
- Πρόσβαση σε ηλεκτρονικές βιβλιοθήκες.
- Δημιουργία ερευνητικών ιστοσελίδων με δημόσια και ιδιωτικά τμήματα.
- Υπογεγραμμένο Λογισμικό.
- Ασφαλές Ηλεκτρονικό Εμπόριο.

1.4 Η δομή της πτυχιακής εργασίας.

Στη παρούσα εργασία παρουσιάζεται καταρχήν μια γενική επισκόπηση και συζήτηση για την κρυπτογραφία και τη χρήση της στην επιστήμη της πληροφορικής. Συγκεκριμένα στο πρώτο κεφάλαιο, το οποίο είναι και εισαγωγικό, γίνεται μια ιστορική αναδρομή σχετικά με την κρυπτογραφία. Εξηγούνται οι σπουδαιότεροι όροι της κρυπτογραφίας και γίνεται συζήτηση σχετικά με την κρυπτογραφία και την εφαρμογή της στην πληροφορική, αλλά και τις υπηρεσίες που μπορεί αυτή να προσφέρει.

Στο δεύτερο κεφάλαιο, εξηγούνται και αναλύονται τα είδη της κρυπτογραφίας. Γίνεται μια σύγκριση της λειτουργίας, αλλά και της λειτουργικότητας, μεταξύ της συμμετρικής και της ασύμμετρης κρυπτογραφίας και διεξάγονται χρήσιμα συμπεράσματα.

Στο τρίτο κεφάλαιο της εργασίας παρουσιάζονται οι βασικές έννοιες και τα θεωρήματα της θεωρίας των αριθμών, που είναι απαραίτητα για την μετέπειτα κατανόηση των περισσοτέρων αλγορίθμων κρυπτογράφησης, αλλά και της θεωρίας και της χρήσης των ελλειπτικών καμπυλών στην κρυπτογραφία.

Στο τέταρτο κεφάλαιο περιγράφονται οι σπουδαιότεροι αλγόριθμοι συμμετρικής κρυπτογραφίας. Συγκεκριμένα αναλύονται ο αλγόριθμος του Καίσαρα, ο αλγόριθμος DES, ο αλγόριθμος IDEA και ο αλγόριθμος AES. Γίνεται μια ιστορική αναδρομή του κάθε αλγορίθμου, περιγράφεται ο αλγόριθμος και αναλύονται οι ιδιότητες και η λειτουργικότητα τους ξεχωριστά για τον καθένα από αυτούς.

Στο πέμπτο κεφάλαιο περιγράφονται οι σπουδαιότεροι αλγόριθμοι ασύμμετρης κρυπτογραφίας. Συγκεκριμένα αναλύονται το πρωτόκολλο ανταλλαγής κλειδιού του Diffie-Hellman, ο αλγόριθμος δημόσιου κλειδιού RSA, η λειτουργία των hash συναρτήσεων και ο αλγόριθμος ElGamal, ο οποίος χρησιμοποιεί το πρόβλημα του διακριτού λογαρίθμου. Γίνεται και εδώ μια ιστορική αναδρομή του κάθε αλγορίθμου, περιγράφεται ο αλγόριθμος και αναλύονται οι ιδιότητες και η λειτουργικότητα τους ξεχωριστά για τον καθένα από αυτούς.

Στο έκτο κεφάλαιο γίνεται μια εισαγωγή στη θεωρία των ελλειπτικών καμπυλών. Εδώ γίνεται τόσο μια γεωμετρική προσέγγιση, όσο και μια αλγεβρική προσέγγιση των ελλειπτικών καμπυλών. Επίσης, αφού εξηγηθούν οι κανόνες που ισχύουν στις ελλειπτικές καμπύλες πραγματικών αριθμών, γίνεται προσαρμογή αυτών σε ελλειπτικές καμπύλες συγκεκριμένων ομάδων αριθμών. Συγκεκριμένα ασχοληθήκαμε με τα σώματα F_p και F_{2^m}. Στο κεφάλαιο αυτό εξηγείται πως γίνονται οι διάφορες πράξεις στα σώματα αυτά και παρουσιάζονται αναλυτικά παραδείγματα για εξάσκηση πάνω σε αυτές. Επίσης, παρουσιάζεται αναλυτικά το πρόβλημα του διακριτού λογαρίθμου και το πως βρίσκει αυτό εφαρμογή στις ελλειπτικές καμπύλες. Τέλος, στο συγκεκριμένο κεφάλαιο γίνεται και μια σύγκριση της λύσης του προβλήματος του διακριτού λογαρίθμου με τη χρήση των ελλειπτικών καμπυλών και της αντίστοιχης λύσης χωρίς τη χρήση αυτών.

Στο έβδομο κεφάλαιο γίνεται παρουσίαση των σπουδαιότερων εφαρμογών των ελλειπτικών καμπυλών στην κρυπτογραφία. Εδώ παρουσιάζονται και εξηγούνται αναλυτικά ορισμένοι αλγόριθμοι κρυπτογραφικών εφαρμογών με τη χρήση των ελλειπτικών καμπυλών. Συγκεκριμένα αυτοί είναι: Ο αλγόριθμος δημιουργίας ψηφιακής υπογραφής με τη χρήση ελλειπτικών καμπυλών (ECDSA), ο αλγόριθμος ανταλλαγής κλειδιού με τη χρήση ελλειπτικών καμπυλών των Diffie - Hellman (ECDH) και ο ολοκληρωμένος αλγόριθμος κρυπτογράφησης – αποκρυπτογράφησης με τη χρήση ελλειπτικών καμπυλών (ECIES).

Τέλος, στο όγδοο κεφάλαιο παρουσιάζονται τα συμπεράσματα τα οποία βγήκαν ολοκληρώνοντας τη παρούσα εργασία και στο τελευταίο κεφάλαιο παρατίθεται η βιβλιογραφία που χρησιμοποιήθηκε, αφού προηγηθεί το παράρτημα της εργασίας.

ΚΕΦΑΛΑΙΟ 2

2. Τα είδη της Κρυπτογραφίας.

2.1 Συμμετρική Κρυπτογραφία.

2.1.1 Έννοια της Συμμετρικής Κρυπτογραφίας.

Στη Συμμετρική Κρυπτογραφία η πληροφορία κωδικοποιείται και αποκωδικοποιείται με το ίδιο μυστικό κλειδί το οποίο πρέπει να γνωρίζουν μόνο εκείνοι που θα έχουν πρόσβαση στην πληροφορία. Ο αποστολέας κρυπτογραφεί το μήνυμα με βάση αυτό το κλειδί και ο παραλήπτης το αποκρυπτογραφεί με βάση το ίδιο κλειδί. Αν τα δύο επικοινωνούντα μέρη βρίσκονται σε διαφορετικές τοποθεσίες, τότε θα πρέπει με κάποιον τρόπο να ανταλλάξουν το κοινό κλειδί που θα πρέπει να χρησιμοποιήσουν. Αυτό ενέχει τον κίνδυνο να υποκλαπεί το κλειδί από κάποιον τρίτο που παρακολουθεί τις γραμμές επικοινωνίας ή και να διαρρεύσει από το ένα από τα δύο μέρη. Αν παραβιαστεί η μυστικότητα του κλειδιού τότε η πληροφορία παύει να είναι ασφαλής [6].

Ο παραλήπτης του μηνύματος θα πρέπει να γνωρίζει τον αλγόριθμο ή κλειδί (key) ή και κλείδα της κρυπτογράφησης για να μπορέσει να αποκωδικοποιήσει και να διαβάσει το μήνυμα. Στην παραδοσιακή κρυπτογραφία ο αποστολέας και ο παραλήπτης του μηνύματος χρησιμοποιούν το ίδιο (κοινό) κλειδί. Στην κρυπτογραφία αυτού του τύπου, θα πρέπει όλα τα κλειδιά που χρησιμοποιούνται να παραμένουν κρυφά, κάτι που είναι εξαιρετικά δύσκολο στα ανοικτά δίκτυα με πολλούς χρήστες, όπως είναι το Internet. Υπάρχουν διάφοροι αλγόριθμοι συμμετρικής κρυπτογραφίας. Οι πιο σημαντικοί είναι οι DES και AES, τους οποίους θα αναλύσουμε παρακάτω. Τα συστήματα συμμετρικής κρυπτογράφησης προϋποθέτουν την ύπαρξη ενός ασφαλούς καναλιού για την ανταλλαγή των μυστικών κλειδιών. Τέτοια συστήματα έχουν αναπτυχθεί και ήδη χρησιμοποιούνται,

με πιο διαδεδομένο το σύστημα Kerberos, του MIT (Massachusetts Institute of Technology).

2.1.2 Λειτουργία της Συμμετρικής Κρυπτογραφίας.

Η συμμετρική κρυπτογραφία, ή αλλιώς κρυπτογραφία μυστικού κλειδιού, χρησιμοποιείται εδώ και χιλιάδες χρόνια σε διαφορετικές μορφές. Η χρήση της μπορεί να απαιτεί απλούς αλγόριθμους αντικατάστασης, αλλά και πιο πολύπλοκες δομές αλγορίθμων. Στις μέρες μας η εξέλιξη της επιστήμης των μαθηματικών και της πληροφορικής, αλλά και η τεράστια αύξηση της υπολογιστικής δύναμης των ηλεκτρονικών υπολογιστών καθιστούν ικανή τη δημιουργία αλγορίθμων πολύ μεγάλης αποδοτικότητας, που σπάνε πολύ δύσκολα.

Η συμμετρική κρυπτογράφηση είναι πολύ γρήγορη, αλλά παράλληλα είναι και πολύ ευπαθής. Αυτό συμβαίνει επειδή το κλειδί με το οποίο γίνεται η κρυπτογράφηση πρέπει να μοιραστεί σε όλους όσους θέλουν να αποκρυπτογραφήσουν το μήνυμα. Ο αλγόριθμος DES, ο οποίος αποτελεί τον πιο γνωστό και πολυχρησιμοποιημένο αλγόριθμο συμμετρικής κρυπτογράφησης, βρίσκεται ήδη στο τέλος της λειτουργίας του. Παρόλα αυτά, η μελέτη του καθίσταται απαραίτητη για την κατανόηση της λειτουργίας της συμμετρικής κρυπτογράφησης.

Η συμμετρική κρυπτογραφία χρησιμοποιείται εδώ και χιλιάδες χρόνια. Ένας από τους πιο απλούς αλγόριθμους συμμετρικής κρυπτογράφησης είναι ο αλγόριθμος του Καίσαρα (Χρησιμοποιήθηκε από τον Ιούλιο Καίσαρα για να αποκρύψει μηνύματα) [7]. Η λειτουργία του είναι απλή και βασίζεται στην αλλαγή θέσης των χαρακτήρων του μηνύματος προς μία συγκεκριμένη φορά. Περαιτέρω ανάλυση θα γίνει σε επόμενο κεφάλαιο.

2.2 Ασύμμετρη Κρυπτογραφία.

2.2.1 Η Ιστορία της Ασύμμετρης Κρυπτογραφίας.

Η ασύμμετρη κρυπτογραφία γεννήθηκε από την ανάγκη να μοιραστεί το κλειδί κρυπτογράφησης με ασφάλεια σε όλους τους παραλήπτες του κρυπτογραφημένου μηνύματος. Στη συμμετρική κρυπτογραφία ο πιο ασφαλής τρόπος ήταν να γίνει η μετάδοση του κλειδιού στους παραλήπτες από κοντά (πρόσωπο με πρόσωπο).

Το 1970 στην Αγγλία, ο James Ellis, ο Clifford Cocks και ο Malcolm Williamson μίλησαν για πρώτη φορά για ασύμμετρη κρυπτογραφία. Το 1976 ο Whitfield Diffie και ο Martin Hellman μίλησαν για την ανταλλαγή κλειδιών σε ένα ασφαλές κανάλι και το 1977 οι Rivest, Shamir και Adleman δημιούργησαν το ασύμμετρο κρυπτογραφικό σύστημα RSA που βασίζεται στη παραγοντοποίηση μεγάλων ακεραίων. Ακολούθησε το κρυπτογραφικό σύστημα του Taher ElGamal το 1985, μέχρι να καταλήξουμε στη θεωρία της κρυπτογραφίας με ελλειπτικές καμπύλες ένα χρόνο αργότερα που αναλύθηκε από τον Neal Koblitz [28].

2.2.2 Έννοια της Ασύμμετρης Κρυπτογραφίας.

Στην ασύμμετρη κρυπτογράφηση ή κρυπτογράφηση δημόσιου κλειδιού χρησιμοποιούμε διαφορετικά κλειδιά για την κρυπτογράφηση και την αποκρυπτογράφηση ενός μηνύματος. Αυτά είναι το **δημόσιο κλειδί (public key)** και το **ιδιωτικό κλειδί (private key)**, τα οποία έχουν τις εξής πολύ σημαντικές ιδιότητες:

- Ένα μήνυμα που έχει κρυπτογραφηθεί με το δημόσιο κλειδί μπορεί να αποκρυπτογραφηθεί μόνο με το αντίστοιχο ιδιωτικό κλειδί και αντίστροφα.
- Αν μας είναι γνωστό το ένα κλειδί δεν μπορούμε να δημιουργήσουμε το άλλο κλειδί.

Η αρχική ιδέα για την κρυπτογράφηση με τη χρήση δημόσιου και ιδιωτικού κλειδιού διατυπώθηκε το 1976 και το 1977 υλοποιήθηκε το κρυπτοσύστημα RSA, που ήταν η πρώτη εφαρμογή ενός συστήματος κρυπτογραφίας που ήταν βασισμένο σε δημόσιο κλειδί [27]. Το δημόσιο κλειδί δεν είναι μυστικό και μπορεί να το αποκτήσει ο οποιοσδήποτε ενδιαφέρεται, ενώ το ιδιωτικό κλειδί χρησιμοποιείται μόνο από τον κάτοχό του και δεν κοινοποιείται σε κανέναν άλλον.

2.2.3 Λειτουργία της Ασύμμετρης Κρυπτογραφίας.

Ο κάθε χρήστης κατέχει ένα ζεύγος κλειδιών, δημόσιο και ιδιωτικό, και όταν στέλνει ένα μήνυμα κωδικοποιημένο με το ιδιωτικό του κλειδί, το μήνυμα αυτό θα μπορεί να αποκωδικοποιηθεί από οποιονδήποτε γνωρίζει το δημόσιο κλειδί του. Έτσι, έχουμε πιστοποίηση του αποστολέα και ακεραιότητα του μηνύματος.

Ενώ όταν ένας χρήστης στέλνει ένα μήνυμα κωδικοποιημένο με το δημόσιο κλειδί του παραλήπτη, το μήνυμα αυτό θα μπορεί να αποκωδικοποιηθεί μόνο με το αντίστοιχο ιδιωτικό του κλειδί του παραλήπτη, οπότε μόνο ο παραλήπτης θα μπορέσει να το διαβάσει και κανένας άλλος και στην περίπτωση αυτή έχουμε εμπιστευτικότητα του μηνύματος.

Η ασύμμετρη κρυπτογράφηση μπορεί να παρέχει πολύ μεγαλύτερη ασφάλεια στις επικοινωνίες σε σχέση με τη συμμετρική κρυπτογράφηση. Αυτή όμως έχει το μειονέκτημα ότι οι αλγόριθμοί της είναι αρκετά βραδύτεροι καθώς απαιτούνται πάρα πολλοί υπολογισμοί.

Τα δύο κλειδιά, ιδιωτικό και δημόσιο, σχετίζονται μαθηματικά μεταξύ τους. Το ένα κρυπτογραφεί και το άλλο αποκρυπτογραφεί το κείμενο. Το ένα από αυτά κρατιέται κρυφό (ιδιωτικό) από το κοινό και το άλλο (δημόσιο) είναι γνωστό σε όλους.

2.3 Συμπεράσματα.

2.3.1 Σύγκριση Συμμετρικής-Ασύμμετρης Κρυπτογραφίας.

Τόσο η συμμετρική όσο και η ασύμμετρη κρυπτογράφηση έχουν τα πλεονεκτήματα και τα μειονεκτήματά τους. Συγκεκριμένα, τα ασύμμετρα κρυπτοσυστήματα είναι πολύ πιο αργά, αλλά παρέχουν μεγαλύτερη ασφάλεια. Στη συμμετρική κρυπτογραφία πρέπει το δημόσιο κλειδί να μοιραστεί στους χρήστες και παράλληλα να μείνει κρυφό από τους αντιπάλους. Αν το κλειδί πέσει σε λάθος χέρια τότε αυτόματα η ασφάλεια των κρυπτογραφημένων μηνυμάτων καταρρίπτεται. Επομένως, στη συμμετρική κρυπτογραφία η ασφαλής μεταφορά του δημόσιου κλειδιού στους πιστοποιημένους χρήστες αποτελεί πολλές φορές σημαντικότερο θέμα επίλυσης και από την ασφαλή κρυπτογράφηση του μηνύματος.

Το μήκος των κλειδιών που χρησιμοποιούνται στη συμμετρική κρυπτογράφηση είναι της τάξης των 56-bit ή 128-bit, τα οποία είναι πολύ μικρότερα των αντίστοιχων της ασύμμετρης κρυπτογραφίας. Ο Phil Zimmermann, δημιουργός του λογισμικού πακέτου για την ασφάλεια Pretty Good Privacy (PGP), αναφέρει ότι ένα 80-bit συμμετρικό κλειδί ισοδυναμεί περίπου με ένα 1024-bit ασύμμετρο κλειδί. Επίσης, αναφέρει ότι για να επιτύχουμε την ασφάλεια που μας προσδίδει ένα 128-bit συμμετρικό κλειδί θα πρέπει να χρησιμοποιήσουμε ένα ασύμμετρο κλειδί μήκους 3000-bit [7].

Η κρυπτογράφηση με μεθόδους συμμετρικής κρυπτογράφησης είναι πολύ πιο γρήγορη από την αντίστοιχη της ασύμμετρης κρυπτογράφησης. Γι' αυτό το λόγο η συμμετρική κρυπτογράφηση χρησιμοποιείται για την κρυπτογράφηση μεγάλου μήκους κειμένου, το οποίο θέλουμε όμως να μείνει κρυφό από τους αντιπάλους για μικρό χρονικό διάστημα. Για παράδειγμα η κρυπτογράφηση με τον αλγόριθμο DES, τον οποίο θα αναλύσουμε σε επόμενο κεφάλαιο είναι 100 φορές πιο γρήγορη από την αντίστοιχη με τον αλγόριθμο ασύμμετρης κρυπτογραφίας RSA σε έναν σύγχρονο επεξεργαστή και 10.000 φορές πιο γρήγορη όταν υλοποιηθεί σε ειδικό επεξεργαστή.

Το μήκος του κλειδιού είναι ένας από τους παράγοντες που καθορίζουν τον βαθμό της ασφάλειας που προσφέρει ο αλγόριθμος κρυπτογράφησης. Το ζητούμενο είναι να εξισορροπηθεί η ασφάλεια που προσφέρει το συγκεκριμένο κρυπτογραφικό σύστημα με το κόστος, τον χρόνο και τα έξοδα υλοποίησης του. Για παράδειγμα ένα συμμετρικό κρυπτογραφικό σύστημα που βασίζεται σε ένα κλειδί μήκους 56-bit είναι μη ασφαλές, αλλά είναι τόσο οικονομικό που μπορεί να χρησιμοποιηθεί από απλούς χρήστες που δεν χρειάζεται να διαφυλάξουν πολύτιμα δεδομένα, τα οποία μάλιστα δεν είναι δυνατόν να διαφυλαχτούν από αποφασισμένους εισβολείς.

2.3.2 Γενικά συμπεράσματα χρήσης Συμμετρικής-Ασύμμετρης Κρυπτογραφίας.

Οι δημιουργοί των κρυπτογραφικών πακέτων και συστημάτων κάνουν αποτίμηση του κόστους υλοποίησης, της ταχύτητας εκτέλεσης της κρυπτογράφησης, των εξόδων για τα πνευματικά δικαιώματα και της ασφάλειας που χρειάζεται πριν την υλοποίηση τους. Αυτό σημαίνει ότι τα κρυπτογραφικά συστήματα δημιουργούνται με την λογική της «όσο το δυνατόν μεγαλύτερη ασφάλεια σε σχέση με το κόστος και την διάρκεια ζωής της εφαρμογής». Μερικές εφαρμογές δεν απαιτούν μεγάλη ασφάλεια με συνέπεια να μη χρειάζεται να σπαταλήσουμε για αυτούς χρόνο και χρήμα για να τις προστατεύσουμε. Ισχυρότεροι υπολογιστές μπορούν να επεξεργαστούν γρηγορότερα μεγαλύτερα κλειδιά, αλλά παράλληλα μπορούν και να σπάσουν ευκολότερα και γρηγορότερα μικρά κλειδιά.

Η κρυπτογραφία δημόσιου κλειδιού, όπως αλλιώς λέγεται η ασύμμετρη κρυπτογραφία, χρησιμοποιείται πιο πολύ στη πιστοποίηση και στη δημιουργία ψηφιακών υπογραφών παρά στην κρυπτογράφηση απλού κειμένου.

ΚΕΦΑΛΑΙΟ 3

3. Βασικές έννοιες της Θεωρίας Αριθμών.

Στο κεφάλαιο αυτό θα περιγραφούν και θα εξηγηθούν οι σημαντικότερες έννοιες της θεωρίας των αριθμών. Πάνω σε αυτές τις έννοιες βασίζονται και οι περισσότεροι κρυπτογραφικοί αλγόριθμοι. Για αυτό το λόγο και το παρακάτω μαθηματικό υπόβαθρο θα πρέπει να συζητηθεί επαρκώς, ώστε να γίνει κατανοητή η ανάλυση των περισσοτέρων αλγορίθμων στα επόμενα κεφάλαια της εργασίας.

3.1 Η πράξη modulo.

Έστω n ένας φυσικός αριθμός και x_1, x_2 δύο ρητοί αριθμοί των οποίων οι παρονομαστές είναι σχετικά πρώτοι (relatively prime) με το n. Δύο αριθμοί είναι **σχετικά πρώτοι** όταν ο μέγιστος κοινός διαιρέτης τους είναι η μονάδα. Ο τελεστής ισοδυναμίας mod για τα x_1, x_2 ορίζεται ως εξής:

$$x_1 = x_2 \pmod{n}$$

Η ισότητα ισχύει αν και μόνο αν:

- $x_1 - x_2 = \alpha / \beta$ με α, β ακέραιους
- gcd (α, β) = 1 (μέγιστος κοινός διαιρέτης)
- α διαιρείται από το n

Παρατηρείται ότι αν $\beta = 1$ και x_1, x_2 είναι ακέραιοι τότε η πράξη mod είναι η οικεία πράξη που όλοι γνωρίζουμε, η οποία δίνει το υπόλοιπο της διαίρεσης του x_1 με το n. Ισχύει $0 \le x_2 \le n-1$.

Ως **αντίστροφος** ενός αριθμού x_1, $x_1^{-1} \pmod{n}$, ορίζεται ο αριθμός ο οποίος αν πολλαπλασιαστεί με τον x_1 δίνει 1(mod n). Για την κατανόηση της πράξης modulo δίνονται τα παρακάτω παραδείγματα [11]:

25 = 4 (mod 7) (γιατί 25 - 4 = 21 = $3 \cdot 7$)

-5 = 3 (mod 4) (γιατί -5 -3 = -8 = $-2 \cdot 4$)

$6^{-1} = 1 \pmod 5$ (γιατί $1 \cdot 6 = 6$ και $6 = 1 \pmod 5$)

$\frac{2}{5} = 1 (\mod 3)$ (γιατί $\frac{2}{5} - 1 = \frac{-3}{5} = 3 \cdot \frac{-1}{5}$)

3.2 Ομάδες

3.2.1 Τι είναι ομάδα.

Ομάδα (group) ονομάζουμε ένα σύνολο αριθμών οι οποίοι έχουν μια καθορισμένη αριθμητική πράξη. Οι μοναδικοί κανόνες αριθμητικής που ορίζουν αυτές τις ομάδες αποτελούν τη πηγή δύσκολων προβλημάτων που χρησιμοποιούνται στην κρυπτογραφική ασφάλεια. Δύο ομάδες που χρησιμοποιούνται στην κρυπτογραφία είναι η Z_n, η προσθετική ομάδα των ακεραίων modulo του αριθμού n και η ομάδα Z_p^*, η πολλαπλασιαστική ομάδα ακεραίων modulo του πρώτου αριθμού p [8].

3.2.2 Η ομάδα Z_n.

Η ομάδα Z_n χρησιμοποιεί μόνο ακέραιους από το 0 έως το n – 1. Η βασική της πράξη είναι η πρόσθεση, η οποία καταλήγει ανάγοντας το τελικό αποτέλεσμα στο modulo n. Αυτό είναι το υπόλοιπο του ακεραίου διαιρούμενο κατά n. Ένα πολύ σημαντικό χαρακτηριστικό της αριθμητικής σε μια ομάδα είναι ότι όλοι οι υπολογισμοί δίνουν αριθμούς που βρίσκονται μέσα στην ίδια ομάδα (**κλειστή ομάδα**). Η modulo μείωση κατά n εξασφαλίζει ότι το αποτέλεσμα της άθροισης είναι μεταξύ του 0 και του n – 1.

Η προσθετική ομάδα για παράδειγμα Z_{15} χρησιμοποιεί ακεραίους από το 0 έως το 14. Εδώ παρουσιάζονται κάποια δείγματα πρόσθεσης στο Z_{15}:

(10 + 12) mod 15 = 22 mod 15 = 7

(4 + 11) mod 15 = 15 mod 15 = 0.

Στο Z_{15}, 10 + 12 = 7 και 4 + 11 = 0.

Παρατηρείστε ότι και οι δύο υπολογισμοί έχουν αποτέλεσμα μεταξύ του 0 και του 14.

Οι αντίστροφοι (αντίθετοι) στη πρόσθεση.

Κάθε αριθμός x σε μια προσθετική ομάδα έχει ένα αντίστροφο (αντίθετο) στοιχείο μέσα σε αυτή την ομάδα. Αυτός είναι ένας ακέραιος –x τέτοιος ώστε $x + (-x) = 0$ μέσα στην ομάδα. Στο Z_{15}, -4 = 11 αφού (4 + 11) mod 15 = 15 mod 15 = 0.

Άλλες πράξεις.

Ενώ η πρόσθεση είναι η κύρια πράξη στη προσθετική ομάδα Z_n, και άλλες πράξεις μπορούν να εξαχθούν από την πρόσθεση. Για παράδειγμα, η αφαίρεση x – y μπορεί να παρουσιαστεί σαν πρόσθεση του x + (-y) mod n.

Στο Z_{15} για παράδειγμα είναι: 1 - 4 = 1 + (-4) = 1 + 11 mod 15 = 12.

Επίσης, μπορεί να ορισθεί ο πολλαπλασιασμός στο Z_n μετά από συνεχόμενες προσθέσεις. Για παράδειγμα, ο πολλαπλασιασμός $4 \cdot 9$ στο Z_{15} μπορεί να επιτευχθεί προσθέτοντας το 9 + 9 + 9 + 9 mod 15 = 36 mod 15 = 6.

3.2.3 Η ομάδα Z_p^*.

Τα κρυπτοσυστήματα χρησιμοποιώντας την αριθμητική στο Z_p^* δημιούργησαν το Πρωτόκολλο Ανταλλαγής Κλειδιού των Diffie - Hellman και τον Αλγόριθμο της Ψηφιακής Υπογραφής (DSA).

Η πολλαπλασιαστική ομάδα Z_p^* χρησιμοποιεί μόνο ακεραίους ανάμεσα στο 1 και το p – 1 (p είναι ένας πρώτος αριθμός), και η βασική της πράξη είναι ο πολλαπλασιασμός. Ο πολλαπλασιασμός ολοκληρώνεται παίρνοντας το υπόλοιπο της διαίρεσης με το p. Η πολλαπλασιαστική ομάδα Z_{11}^*, για παράδειγμα, χρησιμοποιεί

τους αριθμούς από το 1 έως το 10. Ο πολλαπλασιασμός στο Z_{11}^* τελειώνει παίρνοντας το υπόλοιπο όταν το αποτέλεσμα διαιρείται με το 11. Εδώ παρουσιάζονται κάποια παραδείγματα πολλαπλασιασμού στο Z_{11}^*:

$4 \cdot 7 \bmod 11 = 28 \bmod 11 = 6$ και $9 \cdot 5 \bmod 11 = 45 \bmod 11 = 1$.

Έτσι στο Z_{11}^*, $4 \cdot 7 = 6$ και $9 \cdot 5 = 1$.

Παρατηρείστε ότι και οι δύο υπολογισμοί έχουν αποτελέσματα μεταξύ του 1 και του 10.

Οι αντίστροφοι στον πολλαπλασιασμό.

Κάθε αριθμός x σε μια πολλαπλασιαστική ομάδα έχει κάποιο στοιχείο στην ομάδα ως πολλαπλασιαστικό αντίστροφο. Αυτό είναι κάποιος ακέραιος αριθμός x^{-1} τέτοιος ώστε $x \cdot x^{-1} = 1$ στην ομάδα. Στο Z_{11}^*, $9^{-1} = 5$ αφού $9 \cdot 5 \bmod 11 = 1$.

Σε μια πολλαπλασιαστική ομάδα, κάθε στοιχείο πρέπει να έχει πολλαπλασιαστικό αντίστροφο. Σκεφτείτε τους ακέραιους modulo 15. Είναι δυνατόν να ορίσουμε τον πολλαπλασιασμό στους αριθμούς από το 1 έως το 14, πάντα τελειώνοντας με αναγωγή στο modulo 15. Με αυτό το σύστημα, ο αριθμός 6 δεν έχει αντίστροφο αφού δεν υπάρχει αριθμός y τέτοιος ώστε $6 \cdot y \bmod 15 = 1$.

3.2.4 Αβελιανές Ομάδες.

Μια αριθμητική πράξη λέγεται ότι είναι αντιμεταθετική αν η σειρά από τα σύμβολα των πράξεων της είναι ασήμαντη. Σε συνηθισμένους αριθμούς, η πρόσθεση και ο πολλαπλασιασμός είναι αντιμεταθετικές πράξεις. Για παράδειγμα, $2 \cdot 9 = 9 \cdot 2$ και $2 + 9 = 9 + 2$. Ωστόσο, η αφαίρεση και η διαίρεση δεν είναι αντιμεταθετικές πράξεις αφού $2 - 9 \neq 9 - 2$ και $2 / 9 \neq 9 / 2$.

Μια ομάδα λέγεται **αβελιανή** *αν η βασική της πράξη είναι αντιμεταθετική.* Έτσι μια προσθετική ομάδα λέγεται αβελιανή αν ισχύει:

a + b = b + a για όλα τα στοιχεία a, b που ανήκουν στην ομάδα.

Η προσθετική ομάδα Z_n και η πολλαπλασιαστική ομάδα Z_p^* είναι και οι δύο αβελιανές ομάδες.

3.3 Πεπερασμένα Σώματα.

3.3.1 Τι είναι το σώμα.

Σώμα *ονομάζεται ένα σύνολο από στοιχεία με δύο ορισμένες αριθμητικές πράξεις μέσα σε αυτό: πιο συχνά είναι, η πρόσθεση και ο πολλαπλασιασμός.* Τα στοιχεία του σώματος είναι μια προσθετική αβελιανή ομάδα, και τα μη μηδενικά στοιχεία του σώματος αποτελούν μια πολλαπλασιαστική αβελιανή ομάδα. Αυτό σημαίνει ότι όλα τα στοιχεία του σώματος έχουν αντίθετο, και όλα τα μη μηδενικά στοιχεία της ομάδας έχουν πολλαπλασιαστικό αντίστροφο. Όπως είναι γνωστό στις ομάδες, οι υπόλοιπες πράξεις στο σώμα μπορούν να οριστούν στο σώμα, χρησιμοποιώντας τις δύο αυτές κύριες πράξεις.

Ένα σώμα ονομάζεται **πεπερασμένο** *όταν έχει ένα πεπερασμένο αριθμό στοιχείων.* Τα πιο γνωστά πεπερασμένα σώματα δεδομένης τάξης, που χρησιμοποιούνται στην κρυπτογραφία είναι τα σώματα F_p (όπου p ένας πρώτος αριθμός) και F_{2^m} [8].

3.3.2 Το σώμα F_p.

Το πεπερασμένο σώμα F_p (το p είναι ένας πρώτος αριθμός) αποτελείται από τους αριθμούς από το 0 μέχρι το p – 1. Οι πράξεις του σώματος είναι η πρόσθεση και ο πολλαπλασιασμός. Οι πράξεις αυτές ορίζονται για τις ομάδες Z_n και Z_p^* αντίστοιχα: όλοι οι υπολογισμοί ολοκληρώνονται με αναγωγή στο modulo p. Ο περιορισμός ώστε το p να είναι πρώτος αριθμός είναι απαραίτητος έτσι ώστε όλα τα μη μηδενικά στοιχεία να έχουν πολλαπλασιαστικό αντίστροφο (βλέπε Z_p^* για λεπτομέρειες). Όπως με τα Z_n και Z_p^*, οι άλλες πράξεις στο F_p (όπως η διαίρεση, η αφαίρεση και η εκθετοποίηση) προέρχονται από τους ορισμούς της πρόσθεσης και του πολλαπλασιασμού.

Οι υπολογισμοί στο σώμα F_{23}, για παράδειγμα, είναι:

$$10 \cdot 4 - 11 \bmod 23 =$$
$$= 29 \bmod 23 = 6$$

$$7^{-1} \bmod 23 =$$
$$= 10$$

αφού

$$7 \cdot 10 \bmod 23 =$$
$$= 70 \bmod 23 = 1$$

$$8^3/_7 \bmod 23 =$$
$$= 512/_7 \bmod 23 =$$
$$= 6 \cdot 7^{-1} \bmod 23 =$$
$$= 6 \cdot 10 \bmod 23 = 14$$

3.3.3 Το σώμα $F_2{}^m$.

Παρόλο που η περιγραφή του σώματος F_{2^m} είναι πολύπλοκη, αυτό το σώμα είναι πάρα πολύ χρήσιμο διότι οι υπολογισμοί σε αυτό μπορούν να γίνουν αποτελεσματικά όταν υλοποιούνται στους υπολογιστές. Υπάρχουν αρκετοί τρόποι για να περιγραφεί η αριθμητική στο σώμα F_{2^m}. Η **πολυωνυμική αναπαράσταση** περιγράφεται στο επόμενο κεφάλαιο.

3.4 Αναπαραστάση σωμάτων.

3.4.1 Πολυωνυμική Αναπαράσταση.

Τα στοιχεία του σώματος F_{2^m} είναι πολυώνυμα βαθμού μικρότερου του m, με συντελεστές στο σώμα F_2. Έτσι είναι, $\{\mathbf{a_{m-1}x^{m-1} + a_{m-2}x^{m-2} + ... + a_2x^2 + a_1x + a_0 \mid a_i = 0 \text{ ή } 1}\}$. Αυτά τα στοιχεία μπορούν να γραφούν σε μορφή διανύσματος ως εξής: $\mathbf{(a_{m-1} ... a_1 a_0)}$. Το σώμα F_{2^m} έχει 2^m στοιχεία.

Οι βασικές πράξεις στο σώμα F_{2^m} είναι η πρόσθεση και ο πολλαπλασιασμός. Μερικοί υπολογισμοί περιλαμβάνουν το πολυώνυμο $\mathbf{f(x) = x^m + f_{m-1}x^{m-1} + f_{m-2}x^{m-2} + ... + f_2x^2 + f_1x + f_0}$, όπου κάθε $\mathbf{f_i}$ είναι στο σώμα F_2. Το πολυώνυμο f(x) πρέπει να είναι μη αναγόμενο, δηλαδή δε μπορεί να παραγοντοποιηθεί σε 2 μικρότερα πολυώνυμα στο F_2, όπου το καθένα είναι μικρότερο από **m**.

Πρόσθεση

$\mathbf{(a_{m-1} ... a_1 a_0) + (b_{m-1} ... b_1 b_0) = (c_{m-1} ... c_1 c_0)}$ όπου το $\mathbf{c_i = a_i + b_i}$ ανήκει στο F_2. Η πρόσθεση είναι θεωρητικά το XOR του $\mathbf{(a_{m-1} ... a_1 a_0)}$ και $\mathbf{(b_{m-1} ... b_1 b_0)}$.

Αφαίρεση

Στο σώμα F_{2^m}, κάθε στοιχείο $(a_{m-1} ... a_1 a_0)$ έχει σαν συμμετρικό τον ίδιο τον εαυτό του, αφού ισχύει $\mathbf{(a_{m-1} ... a_1 a_0) + (a_{m-1} ... a_1 a_0) = (0 ... 0 0)}$. Έτσι η πρόσθεση και η αφαίρεση είναι ισοδύναμες πράξεις στο F_{2^m}.

Πολλαπλασιασμός

Έχουμε $(a_{m-}1 \ldots a_1\ a_0)\ (b_{m-1} \ldots b_1\ b_0) = (r_{m-1} \ldots r_1\ r_0)$ όπου $r_{m-1}x^{m-1} + \ldots + r_1x + r_0$ είναι το υπόλοιπο του πολυωνύμου $(a_{m-1}x^{m-1} + \ldots + a_1x + a_0)\ (b_{m-1}x^{m-1} + \ldots + b_1x + b_0)$ διαιρούμενο με το πολυώνυμο f(x) στο σώμα F_2. (Σημειώστε ότι όλοι οι συντελεστές του πολυωνύμου είναι modulo 2).

Εκθετοποίηση

Η εκθετοποίηση $(a_{m-1} \ldots a_1\ a_0)^e$ παρουσιάζεται πολλαπλασιάζοντας μαζί **e** αντίγραφα του $(a_{m-1} \ldots a_1\ a_0)$.

Αντίστροφο του πολλαπλασιασμού

Υπάρχει τουλάχιστον ένα στοιχείο g στο σώμα F_{2^m} τέτοιο ώστε όλα τα μη μηδενικά στοιχεία του F_{2^m} μπορούν να εκφραστούν σαν δύναμη του **g**. Ένα τέτοιο στοιχείο **g** λέγεται γεννήτορας του F_{2^m}. Το αντίστροφο του πολλαπλασιασμού ενός στοιχείου $a = g^i$ είναι $a^{-1} = g^{(-i)} \bmod (2m-1)$.

3.4.2 Παράδειγμα Πολυωνυμικής Αναπαράστασης (στο F_2^4).

Τα στοιχεία του F_{2^4} είναι παρακάτω 16 διανύσματα:

(0000) (0001) (0010) (0011) (0100) (0101) (0110) (0111)
(1000) (1001) (1010) (1011) (1100) (1101) (1110) (1111).

Το μη αναγόμενο πολυώνυμο θα είναι το $\mathbf{f(x) = x^4 + x + 1}$. Τα επόμενα είναι δείγματα υπολογισμών.

Πρόσθεση

(0110) + (0101) = (0011).

Πολλαπλασιασμός

(1101) (1001)
$= (x^3 + x^2 + 1)\,(x^3 + 1) \bmod f(x)$
$= x^6 + x^5 + 2x^3 + x^2 + 1 \bmod f(x)$
$= x^6 + x^5 + x^2 + 1 \bmod f(x)$ (οι συντελεστές μειώνονται κατά modulo 2)
$= (x^4 + x + 1)(x^2 + x) + (x^3 + x^2 + x + 1) \bmod f(x)$
$= x^3 + x^2 + x + 1$ (Διαιρέσαμε το $x^6 + x^5 + x^2 + 1$ με το $x^4 + x + 1$)
= (1111).

Εκθετοποίηση

Για να υπολογίσεις το $\mathbf{(0010)^5}$, αρχικά βρίσκεις το
$(0010)^2$
= (0010) (0010)
= x x mod f(x)
$= (x^4 + x + 1)(0) + (x^2) \bmod f(x)$
$= x^2$
= (0100).

Μετά

$(0010)^4$

$= (0010)^2 (0010)^2$

$= (0100) (0100)$

$= x^2 x^2 \bmod f(x)$

$= (x^4 + x + 1)(1) + (x + 1) \bmod f(x)$

$= x + 1$

$= (0011).$

Τελικά, $(0010)^5$

$= (0010)^4 (0010)$

$= (0011) (0010)$

$= (x + 1) (x) \bmod f(x)$

$= (x^2 + x) \bmod f(x)$

$= (x^4 + x + 1)(0) + (x^2 + x) \bmod f(x)$

$= x^2 + x$

$= (0110).$

Αντίστροφο του πολλαπλασιασμού

Το στοιχείο **g = (0010)** είναι ο **γεννήτορας** της ομάδας. Οι δυνάμεις του g είναι:

$g^0 = (0001)$ $g^1 = (0010)$ $g^2 = (0100)$ $g^3 = (1000)$

$g^4 = (0011)$ $g^5 = (0110)$ $g^6 = (1100)$ $g^7 = (1011)$

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

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

Το συμμετρικό στοιχείο του σώματος είναι το $g^0 = (0001)$. Το αντίστροφο του πολλαπλασιασμού του $g^7 = (1011)$ είναι το $g^{-7 \bmod 15} = g^{8 \bmod 15} = (0101)$. Για να το επιβεβαιώσουμε αυτό, βλέπουμε ότι:

$(1011) (0101) =$

$= (x^3 + x + 1)\,(x^2 + 1) \bmod f(x)$

$= x^5 + x^2 + x + 1 \bmod f(x)$

$= (x^4 + x + 1)(x) + (1) \bmod f(x)$

$= 1$

$= (0001),$

το οποίο είναι το αντίστροφο της πράξης του πολλαπλασιασμού.

3.5 Το Κινέζικο Θεώρημα του υπολοίπου.

Το **Κινέζικο Θεώρημα του Υπολοίπου** (**Chinese Remainder Theorem - CRT**) προσφέρει μία μέθοδο για την επίλυση συστημάτων μιας συγκεκριμένης μορφής, που εμπλέκουν ισοδυναμίες ως προς ακέραιους. Το Κινέζικο Θεώρημα του Υπολοίπου εφαρμόζεται κατά την μέτρηση των σημείων μιας ελλειπτικής καμπύλης.

Το Κινέζικο Θεώρημα υπολοίπου μας παρέχει μια αντιστοιχία μεταξύ ενός συστήματος εξισώσεων modulo, το οποίο είναι ένα σύστημα ανά δύο σχετικά πρώτων moduli (όπως για τους αριθμούς 3, 5, 7) και μιας εξίσωσης modulo με το γινόμενό τους (όπως 105). Έχει δύο κύριες χρήσεις. Ας υποθέσουμε ότι ο ακέραιος n παραγοντοποιείται ως $n= n_1 \cdot n_2 \cdots n_k$, όπου οι παράγοντες n_i είναι ανά δύο σχετικά πρώτοι. Κατά πρώτον, το Κινέζικο θεώρημα υπολοίπων περιγράφει τη δομή του Z_n ως ταυτόσημη με τη δομή του Καρτεσιανού γινομένου $Z_{n_1} \times Z_{n_2} \times ..... \times Z_{n_k}$ με πρόσθεση και πολλαπλασιασμό modulo n_i στην i^n συνιστώσα. Κατά δεύτερον, αυτή η περιγραφή μπορεί να χρησιμοποιηθεί προκειμένου να προκύψουν αποτελεσματικοί αλγόριθμοι, επειδή το να δουλεύει κανείς σε καθένα από τα αλγεβρικά συστήματα Z_{n_i} είναι δυνατό να είναι πιο αποτελεσματικό από το να δουλεύει modulo n.

3.5.1 Ορισμός Κινέζικου Θεωρήματος.

(**Κινέζικο Θεώρημα υπολοίπου**). Θεωρούμε ότι τα $m_1, m_2, ..., m_n$ είναι θετικοί ακέραιοι, ανά δύο πρώτοι αριθμοί μεταξύ τους. Έστω ότι τα $\alpha_1, \alpha_2, ..., \alpha_r$ είναι επίσης ακέραιοι. Τότε το σύστημα ισοδυναμιών:

$$x \equiv a_i \pmod{n_i}, 1 \le i \le r,$$

έχει μία μοναδική λύση modulo $n = n_1 \times n_2 \times \ldots\ldots \times n_r$, η οποία δίνεται από τον τύπο:

$$x = \sum_{i=1}^{r} a_i m_i y_i \bmod n,$$

όπου $m_i = n / n_i = n_1 n_2 \ldots n_{i-1} n_{i+1} \ldots n_k$, $i = 1, 2, \ldots k$

και $y_i = m_i^{-1} \bmod n_i$, για $1 \le i \le r$.

3.5.2 Παράδειγμα εφαρμογής του Κινέζικου Θεωρήματος.

Έστω ότι δίνονται οι παρακάτω ισοδυναμίες

$$x \equiv 2 \pmod 9$$

$$x \equiv 1 \pmod 5$$

$$x \equiv 2 \pmod 4$$

Στη συγκεκριμένη περίπτωση είναι:

$\alpha_1 = 2$, $\alpha_2 = 1$, $\alpha_3 = 2$,

$n = 9 \cdot 5 \cdot 4 = 180$,

$n_1 = 9, n_2 = 5, n_3 = 4$

$m_1 = n_2 \cdot n_3 = 5 \cdot 4 = 20$

$m_2 = n_1 \cdot n_3 = 9 \cdot 4 = 36$

$m_3 = n_1 \cdot n_2 = 9 \cdot 5 = 45$

Θέλουμε να υπολογίσουμε το α mod 180, αφού n = 180. Υπολογίζουμε τους αντίστροφους (χρησιμοποιώντας τον αλγόριθμο Ευκλείδη [3] ή με δοκιμές) των 20, 36 και 45, modulo 9, 5 και 4 αντίστοιχα:

$$20^{-1} \equiv 5 \pmod 9$$

$36^{-1} \equiv 1(\mathrm{mod}\, 5)$

$45^{-1} \equiv 1(\mathrm{mod}\, 4)$

Επομένως, είναι:

$c_1 = 20(5\mathrm{mod}\, 9)$

$c_2 = 36(1\mathrm{mod}\, 5)$

$c_3 = 45(1\mathrm{mod}\, 45)$

και

$$a \equiv 2 \cdot 100 + 1 \cdot 36 + 2 \cdot 45(\mathrm{mod}180)$$
$$\equiv 146(\mathrm{mod}180)$$

είναι η μοναδική λύση συστήματος.

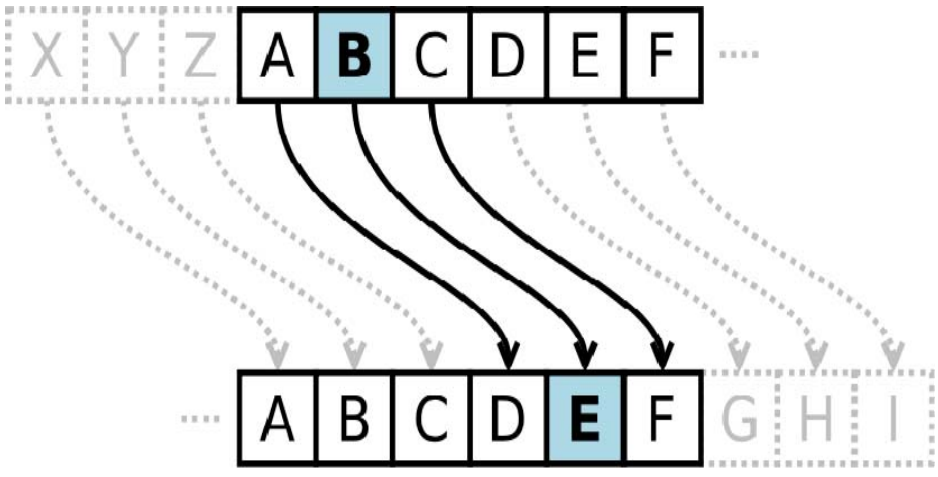
ΚΕΦΑΛΑΙΟ 4

4. Αλγόριθμοι Συμμετρικής Κρυπτογραφίας.

4.1 Ο Αλγόριθμος του Καίσαρα.

4.1.1 Περιγραφή του Αλγόριθμου.

Ο αλγόριθμος του Καίσαρα, ή αλλιώς Αλγόριθμος Αλλαγής Κατεύθυνσης, είναι ο πιο απλός και ο πιο γνωστός αλγόριθμος κρυπτογράφησης. Η ονομασία του αλγόριθμου πάρθηκε από τον Ιούλιο Καίσαρα που τον χρησιμοποίησε για να επικοινωνεί με τα στρατεύματα του μυστικά. Η λειτουργία του βασίζεται στην αντικατάσταση κάθε χαρακτήρα του μηνύματος από έναν άλλο χαρακτήρα, ο οποίος βρίσκεται μερικές συγκεκριμένες θέσεις πιο μακριά από τον πρώτο χαρακτήρα. Για παράδειγμα, αν το κλειδί που θα χρησιμοποιήσουμε είναι το 3 τότε οι θέσεις της μετακίνησης είναι τρεις. Έτσι, το A θα γίνει D, το B θα γίνει E και ούτω καθεξής [29].



Σχήμα 1: Ο Αλγόριθμος του Καίσαρα με κλειδί το 3.

Ο αλγόριθμος του Καίσαρα σπάει πολύ εύκολα, όπως είναι εύκολα κατανοητό. Γι' αυτόν το λόγο, συνήθως το κλειδί δεν είναι ένας ακέραιος, αλλά μία

σειρά από ακέραιους. Έτσι, αν το κλειδί είναι το (5, 19, 1, 2, 11, …), τότε το A θα γίνει E, το B θα γίνει S, το C θα γίνει A, το D θα γίνει B, το E θα γίνει K και ούτω καθεξής. Επίσης, κατά την κρυπτογράφηση θα μπορούσε ένα γράμμα να αντικατασταθεί με ολόκληρη φράση από ένα κείμενο ή το αντίστροφο. Η αποκρυπτογράφηση γίνεται με το ίδιο κλειδί, και πιο συγκεκριμένα με το είδωλο του κλειδιού, παίρνοντας τη σειρά των αριθμών αντίστροφα (το αντίστροφο του αρχικού κλειδιού κρυπτογράφησης) [7].

4.1.2 Παραδείγματα του Αλγόριθμου.

Η μετατροπή του κειμένου σε κρυπτοκείμενο μπορεί να αναπαρασταθεί με δύο αλφάβητα. Το κρυπτο-αλφάβητο προκύπτει από το αλφάβητο αν μετακινήσουμε όλα τα γράμματα κατά κάποιους χαρακτήρες δεξιά ή αριστερά. Αν το κλειδί είναι το 3 θα έχουμε:

Plain: ABCDEFGHIJKLMNOPQRSTUVWXYZ
Cipher: DEFGHIJKLMNOPQRSTUVWXYZABC

Έτσι, κάθε γράμμα στη πρώτη σειρά θα αντικατασταθεί από το αντίστοιχο του στη δεύτερη. Έτσι, στο παρακάτω παράδειγμα θα έχουμε:

Plaintext: the quick brown fox jumps over the lazy dog
Ciphertext: WKH TXLFN EURZQ IRA MXPSV RYHU WKH ODCB GRJ

Επίσης, μπορούμε να χρησιμοποιήσουμε και την modulo αριθμητική για την μετατροπή των χαρακτήρων σε αριθμούς. Αντικαθιστούμε κάθε χαρακτήρα με έναν αριθμό: A=0, B=1, …, Z=25. Η κρυπτογράφηση ενός χαρακτήρα x, με κλειδί το n θα μας δώσει τον y. Συγκεκριμένα για την κρυπτογράφηση έχουμε:

$$E_n(x) = (x + n) \mod 26.$$

και για την αποκρυπτογράφηση:

$$D_n(x) = (x - n) \mod 26.$$

Το αποτέλεσμα είναι πάντα modulo 26 και μας δίνει πάντα ακέραιους αριθμούς από το 0 ως το 25.

4.1.3 Δυνατότητες και σπάσιμο του Αλγόριθμου.

Ο αλγόριθμος μπορεί να σπάσει μέσα σε δευτερόλεπτα με τη μέθοδο της εξαντλητικής αναζήτησης ή αλλιώς brutal force attack, αρκεί ο επιτιθέμενος απλά να υποπτευθεί ότι γίνεται χρήση κρυπτογράφησης με αντικατάσταση. Έχοντας το κρυπτογραφημένο κείμενο δοκιμάζουμε όλα τα κλειδιά ένα ένα μέχρι να δούμε ότι παίρνουμε το αρχικό κείμενο.

Επίσης, μία άλλη μέθοδος αποκρυπτογράφησης και εντοπισμού του κλειδιού κρυπτογράφησης είναι αυτή της ανάλυση σε σχέση με τη συχνότητα Frequency Analysis. Με τη μέθοδο αυτή βλέπουμε ποια γράμματα συναντάμε πιο συχνά στο κρυπτογραφημένο κείμενο. Έτσι, γνωρίζοντας ποια γράμματα γενικά συναντάμε πιο συχνά στην αλφαβήτα (π.χ. το Ε και το Τ συναντιούνται συχνά με μια συγκεκριμένη συχνότητα) μπορούμε να μαντέψουμε ποιο αντιστοιχεί σε ποιο και έτσι να βρούμε το κλειδί κρυπτογράφησης [29]. Στο Παράρτημα Α υπάρχει ο κώδικας υλοποίησης του αλγορίθμου σε java.

4.2 Ο Αλγόριθμος DES (Data Encryption Standard).

4.2.1 Η Ιστορία του Αλγόριθμου.

Η αρχή της δημιουργίας του DES (Data Encryption Standard) ή αλλιώς DEA (Data Encryption Algorithm) χρονολογείται γύρω στο 1970. Συγκεκριμένα, το 1972 το NIST (National Institute of Standards and Technology) στις Ηνωμένες Πολιτείες Αμερικής όρισε διαγωνισμό με σκοπό τη δημιουργία κρυπτογραφικού συστήματος για να προστατευθούν κρίσιμες πληροφορίες του Υπουργείου Εμπορίας της χώρας. Μετά από διάφορες αποτυχημένες προσπάθειες μέσα στο 1973, τον Αύγουστο του 1974 η IBM υπέβαλε υποψηφιότητα με έναν αλγόριθμο, που βασιζόταν στον αλγόριθμο Lucifer του Horst Feistel. Ο Lucifer χρησιμοποιήθηκε το 1970 σε μια εφαρμογή για e-banking. Έτσι, το 1977 η ομάδα της IBM, που αποτελούταν από τους Feistel, Tuchman, Coppersmith, Konheim, Meyer, Matyas, Adler, Grossman, Notz, Smith και Tuckerman, υλοποίησε τη τελική μορφή του DES [30], [31], [32].

Παρακάτω παρουσιάζεται ένας πίνακας με την ιστορία του αλγορίθμου με

χρονολογική σειρά.

Η ζωή του DES μέσα στο χρόνο

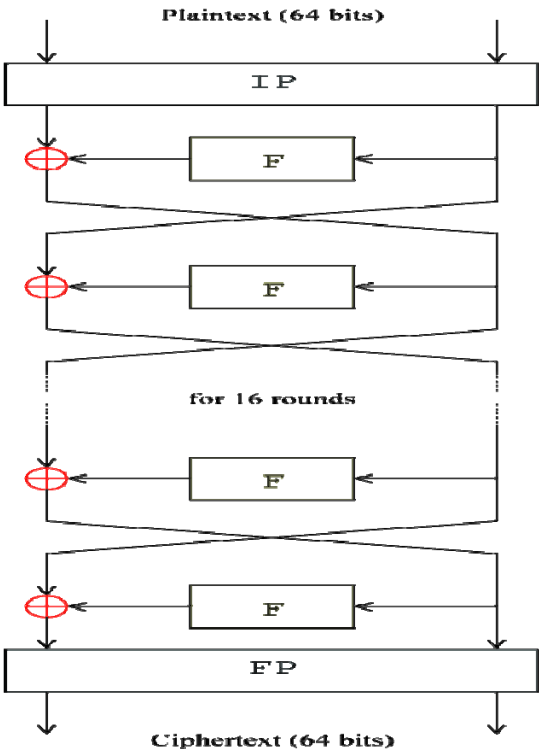
Ημερομηνία	Χρονιά	Γεγονός
15 Μαΐου	1973	Η NIST ζητάει τη δημιουργία κάποιου ικανού κρυπτογραφικού αλγόριθμου
27 Αυγούστου	1974	Η NIST ζητάει ξανά τη δημιουργία κάποιου ικανού κρυπτογραφικού αλγόριθμου, μετά την πρώτη αποτυχημένη απόπειρα
17 Μαρτίου	1975	Ο DES δημοσιεύεται
Αύγουστος	1976	Πρώτη δοκιμή του DES
Σεπτέμβριος	1976	Δεύτερη δοκιμή του DES
Νοέμβριος	1976	Ο DES εγκρίνεται
15 Ιανουαρίου	1977	Ο DES δημοσιεύεται σαν FIPS (Federal Information Processing Standard)
	1983	Ο DES επικυρώνεται σαν κρυπτογραφικός αλγόριθμος
	1986	Ο DES χρησιμοποιείται σαν Videocipher II και σαν σύστημα δορυφορικής παρεμβολής στη τηλεόραση
22 Ιανουαρίου	1988	Ο DES επικυρώνεται σαν κρυπτογραφικός αλγόριθμος για δεύτερη φορά
	1992	Οι Biham and Shamir αναφέρουν την πρώτη θεωρητική επίθεση με τη χρήση της διαφορικής κρυπτανάλυσης
30 Δεκεμβρίου	1993	Επικυρώνεται η ασφάλεια που παρέχει για μια ακόμα φορά

Ιούνιος	1997	Το ερευνητικό έργο DESCHALL μελετάει και σπάει ένα μήνυμα που κρυπτογραφήθηκε με τον DES
Ιούλιος	1998	Η εφαρμογή EFF's DES cracker (Deep Crack) σπάει τον DES σε 56 ώρες
Ιανουάριος	1999	Μαζί τα Deep Crack και distributed.net σπάνε τον DES μέσα σε 22 ώρες και 15 λεπτά
25 Οκτωβρίου	1999	Επικυρώνεται η ασφάλεια που παρέχει για μια ακόμα φορά Ο DES, αλλά αυτή τη φορά προτείνεται η χρήση του 3-DES
26 Νοεμβρίου	2001	Ο AES (Advanced Encryption Standard) δημοσιεύθηκε
26 Μαΐου	2002	Ο AES υλοποιείται
26 Ιουλίου	2004	Η απόσυρση του DES προτείνεται
19 Μαΐου	2005	Ο DES αποσύρεται
15 Μαρτίου	2007	Η συσκευή FPGA, του πανεπιστημίου του Bochum και Kiel στη Γερμανία, σπάει τον DES μέσα σε 6,5 μέρες με κατασκευαστικό κόστος στα $10.000

4.2.2 Περιγραφή του Αλγόριθμου.

Ο DES είναι ο πιο πολυχρησιμοποιημένος αλγόριθμος συμμετρικής κρυπτογράφησης. Αυτός χρησιμοποιήθηκε πάρα πολύ για την προστασία οικονομικών δεδομένων και πιο συγκεκριμένα βρήκε εφαρμογή στα ATMs (Automated Teller Machines) [7]. Ο DES είναι αλγόριθμος που χρησιμοποιεί κλειδί μεγέθους 56-bit. Άλλα 8 bit χρησιμοποιούνται για την ισοτιμία στο τέλος. Έτσι, το μέγεθος του κλειδιού μετατρέπεται σε 64-bit. Εξαιτίας του μικρού μήκους κλειδιού του ο DES δεν αποτελεί πια μια ασφαλή λύση. Το 1999 οι εταιρίες distributed.net

και Electronic Frontier Foundation σε συνεργασία μαζί κατόρθωσαν και βρήκαν ένα κρυπτογραφικό κλειδί του DES μέσα σε 22 ώρες και 15 λεπτά. Γι' αυτό το λόγο έχει αντικατασταθεί πια από τον AES, που θα δούμε σε επόμενη ενότητα.



Σχήμα 2: Η δομή του DES.

Η δομή του αλγορίθμου φαίνεται στο σχήμα 2. Ο αλγόριθμος λειτουργεί ως εξής [32]:

- Παίρνει ένα string από bits και με σύνθετες διεργασίες το μετατρέπει σε ένα διαφορετικό string από bits.
- Το μήκος του string αυτού είναι 64 bits (τα 8 χρησιμοποιούνται σαν bits ισοτιμίας).
- Για να γίνει αυτή η μετατροπή χρησιμοποιείται ένα κλειδί 64άρων bits, από τα οποία μόνο τα 56 χρησιμοποιούνται.
- Ουσιαστικά γίνονται 16 αντιμεταθέσεις των bits μέχρι να πάμε από την αρχική μορφή IP (Initial Permutation) του string από bits στη τελική του μορφή FP (Final Permutation).
- Αν η σειρά των κλειδιών που χρησιμοποιούνται για τη κρυπτογράφηση

είναι {k1, k2, k3,…., k15}, τότε η σειρά των κλειδιών για την αποκρυπτογράφηση θα είναι η αντίστροφη, δηλαδή {k15, k14, k13,…., k1}.

- Αρχικά το block χωρίζεται σε 2 ίσα μέρη των 32 bits.
- Η συνάρτηση F μπερδεύει το μισό block με τη βοήθεια του αντίστοιχου κλειδιού και το αποτέλεσμα συνδυάζετε με το άλλο μισό block με ένα XOR.
- Στο νέο block που προκύπτει από το XOR εφαρμόζουμε το 2ο κλειδί και ξαναμπερδεύεται το block για να συνδυαστεί με τη χρήση XOR με το block των bits που υπήρχε πριν εφαρμοστεί το 1ο κλειδί επάνω του.
- Αυτό γίνεται 15 φορές μέχρι τη τελική αντιμετάθεση.

4.2.3 Γενικά συμπεράσματα για τη χρήση και την ασφάλεια του Αλγόριθμου.

Ο αλγόριθμος χρησιμοποιήθηκε για πάνω από 20 χρόνια κάτι που σημαίνει ότι ήταν πολύ χρήσιμος και εύχρηστος. Στην αρχή η ασφάλεια του ήταν καλή, αλλά με τον καιρό ξεπεράστηκε από καινούργια κρυπτογραφικά συστήματα που χρησιμοποιούσαν νέες τεχνολογίες. Ήταν θέμα χρόνου πια η αποκρυπτογράφηση των μηνυμάτων που κρυπτογραφόντουσαν με τον DES.

Ο DES χρησιμοποιείται ακόμα και σήμερα, αλλά για την κρυπτογράφηση μηνυμάτων, τα οποία θέλουμε να μείνουν κρυφά για μικρό χρονικό διάστημα. Έτσι, αν θέλουμε να κρατήσουμε ένα μήνυμα κρυφό για ορισμένα δευτερόλεπτα (π.χ. μόνο κατά τη μεταφορά του), μπορούμε να τον εφαρμόσουμε. Με λίγα λόγια, ο αλγόριθμος χρησιμοποιείται πια μόνο για τη διασφάλιση της ακεραιότητας των μηνυμάτων και όχι για την μόνιμη απόκρυψή τους. Ο αλγόριθμος χρησιμοποιείται όταν ο χρόνος για την εύρεση του κλειδιού είναι μικρότερος από τον αντίστοιχο χρόνο για την απόκρυψη των δεδομένων.

Το σπάσιμο του DES μπορεί να γίνει με τη μέθοδο της *«εξαντλητικής αναζήτησης» ή αλλιώς Brute force attack*. Με τη μέθοδο αυτή δοκιμάζεται κάθε πιθανό κλειδί μέχρι να βρεθεί το σωστό. Το μήκος του κλειδιού καθορίζει και τον αριθμό των κλειδιών που θα πρέπει να δοκιμαστούν μέχρι να βρούμε το σωστό. Το

σύνολο των κλειδιών που πρέπει να δοκιμαστούν είναι περίπου $2^{56} \approx 72 \cdot 10^{15}$ κλειδιά. Επομένως ήταν θέμα χρόνου και κόστους για να σπάσει ο αλγόριθμος.

Ο χρόνος σε σχέση με το κόστος είναι αντιστρόφως ανάλογα. Όσο γρηγορότερα θέλουμε να βρούμε ένα κλειδί τόσο ισχυρότερος πρέπει να είναι ο επεξεργαστής που θα χρησιμοποιήσουμε, ώστε να κάνει πιο γρήγορα πράξεις, και επομένως και το κόστος κατασκευής του αυξάνεται. Για το σπάσιμο του DES χρειάζονται 10 χρόνια για έναν απλό σύγχρονο επεξεργαστή και 1 απόγευμα για έναν νέο ακριβό επεξεργαστή που θα κατασκευαστεί αποκλειστικά για το σπάσιμο του DES. Ο επεξεργαστής αυτός θα κόστιζε όμως πάνω από 20.000.000 $ το 1997 (Garon – Outbridge). Στον παρακάτω πίνακα φαίνονται οι απαιτήσεις σε δολάρια σε σχέση με το χρονικό διάστημα που θέλουμε να σπάσει ο αλγόριθμος σε διαφορετικά χρονολογικά έτη [32], [33].

Απαιτήσεις σε δολάρια $ για το σπάσιμο του DES				
Έτος	**1 Χρόνος**	**1 Μήνας**	**1 Εβδομάδα**	**1 Ημέρα**
1990	129.000	1.532.000	6.664.000	46.622.000
1995	52.000	600.000	2.611.000	18.265.000
2000	10.300	117.000	510.000	3.580.000

Ο προηγούμενος πίνακας έγινε μετά από υπολογισμούς που έγιναν από τον Garon και τον Outerbridge [33]. Οι τιμές των micro-chips διαφέρουν κάθε χρονολογικό έτος. Έτσι, οι τιμές τους ήταν οι εξής:

- Το 1990, το 1 Mhz chip κόστιζε 25$, το 2 Mhz chip κόστιζε 250$, ενώ το 4 Mhz chip ήταν πολύ ακριβό για την εποχή.
- Το 1995, το 2 Mhz chip κόστιζε 25$, το 4 Mhz chip κόστιζε 250$, ενώ το 32 Mhz chip ήταν πολύ ακριβό για την εποχή.
- Το 2000, το 4 Mhz chip κόστιζε 25$, το 32 Mhz chip κόστιζε 250$, ενώ το 256 Mhz chip ήταν πολύ ακριβό για την εποχή.

4.3 Ο Αλγόριθμος IDEA (International Data Encryption Algorithm).

4.3.1 Η Ιστορία του Αλγόριθμου.

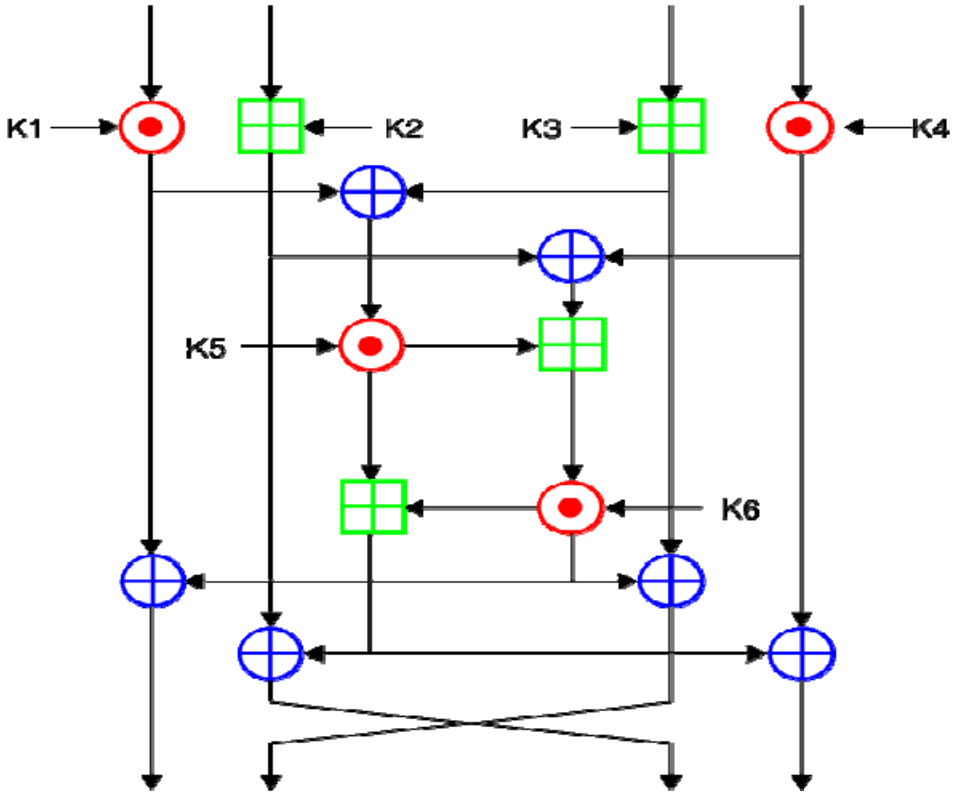
Ο IDEA (International Data Encryption Algorithm) είναι ένας κρυπτογραφικός αλγόριθμος που σχεδιάστηκε από τους Lai X. και Massey J. του Πανεπιστημίου Τεχνολογίας της Ζυρίχης το 1991 [7], [34]. Ο αλγόριθμος σχεδιάστηκε για να αντικαταστήσει τον DES. Ο IDEA αποτελεί μια επανάληψη του αλγόριθμου PES (Proposed Encryption Standard), γι' αυτό και ο IDEA αποκαλείται αλλιώς και IPES (Improved PES).

Ο αλγόριθμος αυτή τη στιγμή είναι ελεύθερος για μη-εμπορική χρήση και τα δικαιώματά του ανήκουν στην εταιρεία MediaCrypt, μέχρι το 2011. Ο IDEA χρησιμοποιήθηκε στην εφαρμογή PGP (Pretty Good Privacy) v2.0. Η εφαρμογή αυτή παρέχει κρυπτογράφηση και πιστοποίηση και χρησιμοποιείται ευρέως για την κρυπτογράφηση και αποκρυπτογράφηση στο ηλεκτρονικό ταχυδρομείο, όπου και παρέχει ασφάλεια. Δημιουργήθηκε το 1991 από τον Philip Zimmermann [35].

4.3.2 Περιγραφή του Αλγόριθμου.

Ο αλγόριθμος IDEA χρησιμοποιεί κλειδιά μεγέθους 128-bit, τα οποία τα εφαρμόζει στο κείμενο σε 8 γύρους. Έτσι, λόγω του μεγαλύτερου μεγέθους κλειδιού από αυτό του DES παρέχει μεγαλύτερη ασφάλεια από αυτόν. Η κρυπτογράφηση γίνεται σε ένα block μήκους 64-bit με τη χρήση κλειδιού μήκους 128-bit. Η κρυπτογράφηση ολοκληρώνεται μετά από 8 γύρους και μια τελευταία μετατροπή που ονομάζεται μισός γύρος.

Η λειτουργία του αλγόριθμου φαίνεται στο σχήμα 3:



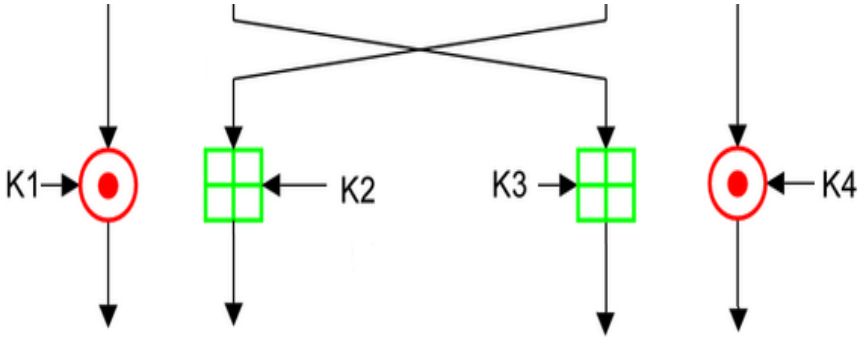
Σχήμα 3: Ένας γύρος από τους 8 συνολικά του IDEA.

Οι πράξεις που λαμβάνουν χώρα είναι οι εξής:

- Η αποκλειστική διάζευξη, το γνωστό XOR (eXclusive OR) (συμβολίζεται με έναν μπλε σταυρό ⊕).
- Η πρόσθεση modulo 2^{16} (συμβολίζεται με έναν πράσινο σταυρό ⊞).
- Ο πολλαπλασιασμός modulo $2^{16}+1$, (συμβολίζεται με μία κόκκινη βούλα ⊙).

Όπως βλέπουμε η είσοδος είναι τετραπλή, δηλαδή η 64-bit ψηφιολέξη χωρίζεται σε 4 ψηφιολέξεις των 16-bit. Το κλειδί K έχει μήκος 128-bit και διασπάται σε 6 κλειδιά (k1, k2, k3, …, k6) των 16-bit. Τα πρώτα 8 bits του κλειδιού αφαιρούνται από το κλειδί από την αρχή. Έτσι, το k1 του πρώτου γύρου αποτελείται από τα πρώτα 16 bits του κλειδιού και κάθε φορά που ξεκινάει ο κάθε γύρος μετακινούνται όλα τα ψηφία του κλειδιού κατά 25 θέσεις αριστερά.

Οι πράξεις αυτές θα γίνουν 8 φορές όπως φαίνεται στο σχήμα. Κατόπιν, θα εφαρμοστεί ο τελευταίος «μισός γύρος», όπως φαίνεται στο σχήμα 4.



Σχήμα 4: Ο μισός τελευταίος γύρος του IDEA.

4.3.3 Ασφάλεια του Αλγόριθμου.

Οι σχεδιαστές όταν δημιούργησαν τον αλγόριθμο, τον ανέλυσαν και μέτρησαν τις αντοχές του σε επιθέσεις με διαφορική κρυπτανάλυση. Τα συμπεράσματα ήταν αρκετά ικανοποιητικά. Το 1994, βρέθηκαν κάποιες ομάδες κλειδιών του αλγορίθμου που ήταν πολύ πιο ευπαθείς από τις άλλες [36]. Όμως, λόγω του μεγάλου μήκους κλειδιού (128-bit) και επομένως και του μεγάλου αριθμού κλειδιών που υπάρχουν (2^{128} κλειδιά), αυτό δεν αποτέλεσε σπουδαίο πρόβλημα. Το 2005 όμως, δημοσιεύτηκε μία εργασία όπου με τη μέθοδο της εξαντλητικής αναζήτησης έβρισκε το κλειδί του αλγορίθμου, ο οποίος τότε υλοποιούταν σε 6 γύρους και όχι σε 8. Έτσι, το 2005 η MediaCrypt δημιούργησε μια νέα έκδοση του αλγόριθμου με μεγαλύτερη ασφάλεια, τον IDEA NTX.

4.4 Ο Αλγόριθμος AES (Advanced Encryption Standard).

4.4.1 Η ιστορία του Αλγόριθμου.

Ο AES (Advanced Encryption Standard) δημιουργήθηκε με την προοπτική να αντικαταστήσει τον DES, όταν αυτός είχε πια φτάσει στο τέλος της ζωής του και όταν πια η ασφάλεια του ξεπεράστηκε. Το 1997, ανακοινώθηκε ένας διαγωνισμός από το NIST (National Institute of Standards and Technology) στις Η.Π.Α, στον

οποίο υπήρξαν 15 υποψηφιότητες. Από αυτές τις 15 μόνο οι 5 εγκρίθηκαν. Τελικά νικητής του διαγωνισμού ανακηρύχτηκε το προϊόν του οποίου δημιουργοί ήταν οι Joan Daemen και Vincent Rijmen από το Βέλγιο. Αυτοί ήταν και οι δημιουργοί του αλγόριθμου με την ονομασία Rijndael [7].

Ο συγκεκριμένος αλγόριθμος είναι αρκετά πολύπλοκος, αλλά η ασφάλεια που παρέχει είναι αρκετά καλή. Επίσης είναι αρκετά γρήγορος. Ο AES χρησιμοποιείται κυρίως στις έξυπνες κάρτες.

4.4.2 Η λειτουργία του Αλγόριθμου.

Η λειτουργία του αλγορίθμου είναι αρκετά πολύπλοκη. Για αυτό τον λόγο θα προσπαθήσουμε να την εξηγήσουμε όσο πιο περιληπτικά, αλλά και κατανοητά γίνεται. Η είσοδος και η έξοδος του αλγορίθμου αποτελείται από 128 bits. Αυτές οι σειρές από bits αναφέρονται συνήθως σαν blocks. Στον συγκεκριμένο αλγόριθμο τα μήκη των κλειδιών που χρησιμοποιούνται είναι 128, 192 ή 256 bits. Τα bits σε κάθε block αριθμούνται από το 0 μέχρι έναν αριθμό μικρότερο του μήκους του block. Η αρίθμηση αυτή ονομάζεται index i [38].

Η βασική μονάδα επεξεργασίας στον AES είναι το byte. 1 byte είναι μια σειρά από 8 συνεχόμενα bits. Έτσι, το κάθε block χωρίζεται σε 8άδες, και συγκεκριμένα σε 128/8 = 16 8άδες. Αυτές οι 16 8άδες σχηματίζουν έναν πίνακα 16 θέσεων. Έστω έχουμε τη πληροφορία 11010100. Αυτή σχηματίζει το πολυώνυμο βαθμού < 8 και συγκεκριμένα το πολυώνυμο $x^7 + x^6 + x^4 + x^2$. Γενικά θα ισχύει:

- Ένα byte θα αποτελείται από τα εξής bits: $\{b_7,\ b_6,\ b_5,\ b_4,\ b_3,\ b_2,\ b_1,\ b_0\}$
- Το πολυώνυμο που θα προκύπτει θα είναι της μορφής:

$$b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0 = \sum_{i=0}^{7} b_i x^i$$

Επίσης, προσδίδουμε τιμές στα bytes με 16αδική αναπαράσταση, σύμφωνα με το σχήμα 5. Έτσι, το byte {01100011} θα αναπαριστάται ως {63}.

Bit Pattern	Character
0000	0
0001	1
0010	2
0011	3

Bit Pattern	Character
0100	4
0101	5
0110	6
0111	7

Bit Pattern	Character
1000	8
1001	9
1010	a
1011	b

Bit Pattern	Character
1100	c
1101	d
1110	e
1111	f

Σχήμα 5: Δεκαεξαδική αναπαράσταση των bytes στο AES.

Αποτέλεσμα όλων αυτών είναι να δημιουργηθεί ένας πίνακας 16 θέσεων, ο οποίος ονομάζεται State. Τα 4 bytes σε κάθε στήλη c του State σχηματίζουν μία «ψηφιολέξη» και ο αριθμός της γραμμής r μας δείχνει τη θέση των 4άρων bytes της λέξης. Έτσι, αντί για πίνακα θα μπορούσαμε να έχουμε και την εξής μορφή:

$w_0w_1w_2w_3$, όπου w η ψηφιολέξη και 0, 1, 2, 3 ο αριθμός της στήλης της κάθε λέξης

Ο αλγόριθμος χρησιμοποιεί σε κάθε στάδιο λειτουργίας του αρκετές μαθηματικές πράξεις οι οποίες θεωρούμε ότι επενεργούν στα στοιχεία του πεπερασμένου σώματος **GF(2^8)**:

Πρόσθεση: Η πρόσθεση είναι η XOR των bits, πχ.

$$\mathbf{(x^6 + x^4 + x^2 + x + 1) + (x^7 + x + 1) =}$$

$$\mathbf{x^7 + x^6 + x^4 + x^2}$$

Πολλαπλασιασμός: Πολλαπλασιασμός στο **GF(2^8)** είναι ο πολλαπλασιασμός μεταξύ πολυωνύμων modulo (ένα ανάγωγο (irreducible) πολυώνυμο βαθμού 8). Ανάγωγο ονομάζεται ένα πολυώνυμο αν διαιρείται μονάχα από τον εαυτό του και την μονάδα. Το πολυώνυμο που έχει επιλεχθεί για το AES είναι το :

$m(x) = \mathbf{x^8 + x^4 + x^3 + x + 1}$

Αρχικά ας πολλαπλασιάσουμε το $x^7 + x^6 + x^3 + x + 1$ με το x :

$(x^7 + x^6 + x^3 + x + 1) \cdot x = x^8 + x^7 + x^4 + x^2 + x$

$\equiv (x^8 + x^7 + x^4 + x^2 + x) \bmod (x^8 + x^4 + x^3 + x + 1)$

$\equiv (x^7 + x^3 + x^2 + 1)$

Η ίδια πράξη με bits γίνεται:

11001011 => 110010110 (ολίσθηση (shift) αριστερά και προσθήκη ενός 0) =>110010110 + 100011011 = 010001101 που αντιστοιχεί στο προηγούμενο αποτέλεσμα.

Ο αλγόριθμος βασίζεται σε συνεχείς αλλαγές των bytes του κειμένου βασισμένες στο κλειδί και στη συνέχεια η αποκρυπτογράφηση βασίζεται πάλι στο ίδιο κλειδί [39], [40], [41].

Στη διαδικασία κρυπτογράφησης υπάρχουν τέσσερις κυκλικές συναρτήσεις:

- SubBytes
- ShiftRows
- MixColumns
- AddRound Key

Ο μετασχηματισμός SubBytes() είναι μια μη γραμμική αντικατάσταση οκτάδων που λειτουργεί ανεξάρτητα σε κάθε οκτάδα της κατάστασης χρησιμοποιώντας έναν πίνακα αντικατάστασης (s-box). Ο πίνακας S-box προκύπτει συνήθως από το κλειδί με διάφορους τρόπους που περιέχουν πολλές μαθηματικές πράξεις [42]. Ένας τρόπος φαίνεται στο σχήμα 6:

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

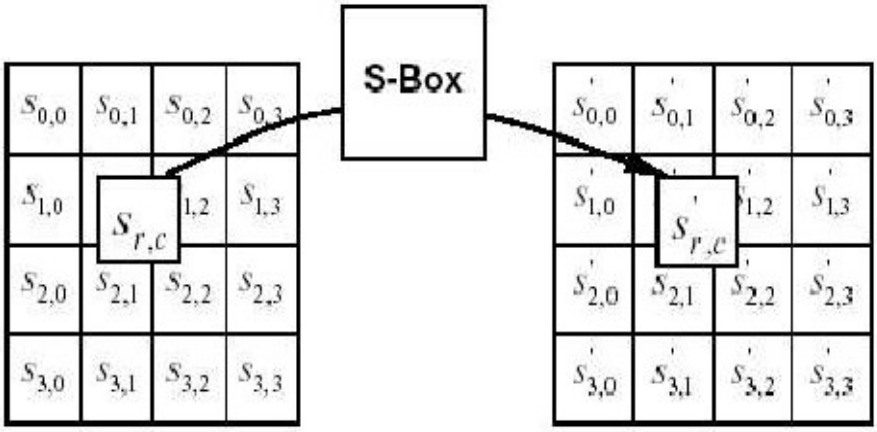
Σχήμα 6: Μαθηματικά από τα οποία προκύπτει κάποιο S-box.

Στο σχήμα 6 είναι b_i το $i^{-ιοστό}$ bit του byte του κλειδιού. Αυτή η πράξη γίνεται και για τις 16 8άδες του κλειδιού και έτσι παράγονται 16 νέες 8άδες, όπως στο σχήμα 7. Έστω, έχουμε το S-box του σχήματος 7.

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

Σχήμα 7: Παράδειγμα S-box.

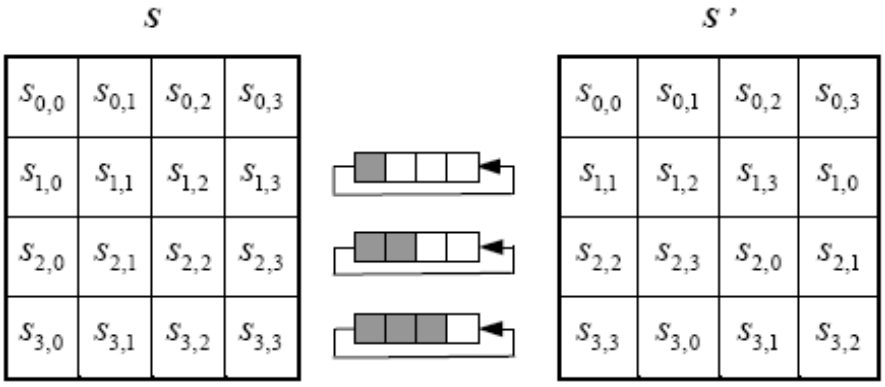
Ο μετασχηματισμός SubBytes() γίνεται ως εξής:



Σχήμα 8: Μετασχηματισμός SubBytes().

Έτσι, για παράδειγμα αν το $s_{1,1} = \{53\}$, τότε η τιμή που θα αντικαταστήσει αυτή τη τιμή θα είναι η τομή της γραμμής 5 με την στήλη 3 του σχήματος 7, δηλαδή το $s'_{1,1} = \{ed\}$.

Στο **μετασχηματισμό ShiftRows()**, οι ψηφιολέξεις στις τελευταίες τρεις σειρές της State μετατοπίζονται κυκλικά πέρα από τους διαφορετικούς αριθμούς ψηφιολέξεων. Η πρώτη σειρά, r = 0, δεν μετατοπίζεται (σχήμα 9).



Σχήμα 9: Μετασχηματισμός ShiftRows().

Στον μετασχηματισμό **MixColumns()** πολλαπλασιάζεται κάθε στήλη με ένα πολυώνυμο c(x) modulo $x^4 + 1$. Κάθε στήλη θεωρείται σαν πολυώνυμο $4^{ου}$ βαθμού.

Στον μετασχηματισμό **AddRoundKey()** ένα "Κλειδί Γύρου" προστίθεται στην State με την χρήση μίας απλής XOR. Το κλειδί γύρου προκύπτει από το αρχικό Κλειδί Κρυπτογράφησης (ή μυστικό κλειδί) με τη μέθοδο Key Schedule. Το μήκος του είναι ίσο με το μήκος του Block (Nb). Ο αλγόριθμος του παραγωγής του κλειδιού γύρου παρουσιάζεται στο παράρτημα Β.

Έτσι, γνωρίζοντας αυτές τις πράξεις και εφαρμόζοντας τον αλγόριθμο του σχήματος 10 πραγματοποιούμε την κρυπτογράφηση.

```
Cipher(byte in[4*Nb], byte out[4*Nb], word w[Nb*(Nr+1)])
begin
   byte  state[4,Nb]

   state = in

   AddRoundKey(state, w[0, Nb-1])                  // See Sec. 5.1.4

   for round = 1 step 1 to Nr-1
      SubBytes(state)                              // See Sec. 5.1.1
      ShiftRows(state)                             // See Sec. 5.1.2
      MixColumns(state)                            // See Sec. 5.1.3
      AddRoundKey(state, w[round*Nb, (round+1)*Nb-1])
   end for

   SubBytes(state)
   ShiftRows(state)
   AddRoundKey(state, w[Nr*Nb, (Nr+1)*Nb-1])

   out = state
end
```

Σχήμα 10: Ψευδοκώδικας του αλγόριθμου AES.

4.4.3 Η ασφάλεια του Αλγόριθμου.

Μέχρι το 2006, οι μοναδικές περιπτώσεις επιτυχών επιθέσεων στον AES είχαν να κάνουν με τις επιθέσεις με «έμμεσο τρόπο» (side-channel attacks). Στις επιθέσεις αυτές ο επιτιθέμενος δεν προσπαθεί να βρει το μυστικό κλειδί με μαθηματικά ή επιλύοντας τον αλγόριθμο δοκιμάζοντας διάφορα κλειδιά, όπως στην εξαντλητική αναζήτηση. Αντίθετα εδώ, οι πληροφορίες ανακτώνται μετά από μελέτη κυρίως της φυσικής υλοποίησης του αλγορίθμου. Για παράδειγμα, πληροφορίες σχετικά με το χρόνο κρυπτογράφησης και την υπολογιστική δύναμη που καταναλώθηκε θα μπορούσαν να δώσουν χρήσιμες πληροφορίες που να οδηγήσουν στην εύρεση του κλειδιού και την αποκρυπτογράφηση των δεδομένων. Έτσι μερικά παραδείγματα επιθέσεων με έμμεσο τρόπο είναι τα εξής [43]:

- **Επίθεση με χρονομέτρηση (timing attack):** Παρακολουθεί και χρονομετρεί την είσοδο και έξοδο των δεδομένων σε έναν επεξεργαστή ή στη μνήμη στα οποία τρέχει ο αλγόριθμος. Επίσης, μπορεί να μετράει το χρόνο που χρειάζεται για να γίνει η κρυπτογράφηση σε διαφορετικά κείμενα. Έτσι, μπορεί να υπολογιστεί το μήκος του κλειδιού.
- **Επίθεση με παρακολούθηση της υπολογιστικής δύναμης (power monitoring attack):** Παρακολουθεί και καταγράφει την υπολογιστική δύναμη, κυρίως του επεξεργαστή που χρησιμοποιείται για την κρυπτογράφηση. Πολλές δοκιμές σε διαφορετικούς επεξεργαστές και με διαφορετικά κρυπτογραφημένα δεδομένα μπορούν να μας οδηγήσουν στην εύρεση του κλειδιού.
- **Επίθεση με θερμική απεικόνιση:** Είναι μια αρκετά καινούργια μέθοδος του Shamir με την οποία οι θερμικές αλλαγές της επιφάνειας του επεξεργαστή απεικονίζονται σαν εικόνες στην οθόνη και μας δείχνουν έτσι πως λειτουργεί ο επεξεργαστής κατά τη διάρκεια της εφαρμογής του αλγορίθμου.

Το 2003, η NSA (National Security Agency) ανακοίνωσε ότι ο AES είναι αρκετά ασφαλής και μπορεί να χρησιμοποιηθεί από την κυβέρνηση των Η.Π.Α. για την προστασία μη-εμπιστευτικών δεδομένων. Συγκεκριμένα εκδόθηκε η εξής ανακοίνωση:

Ο σχεδιασμός και η ισχύς όλων των κλειδιών του AES είναι αρκετές για να

προστατεύσουν εμπιστευτικές πληροφορίες μέχρι το επίπεδο του «ΑΠΟΡΡΗΤΟΥ». Εμπιστευτικές πληροφορίες του επιπέδου «ΑΚΡΩΣ ΑΠΟΡΡΗΤΟ» απαιτούν κλειδιά μήκους 192 ή 256 bits. Οι διάφορες υλοποιήσεις του AES μπορούν να προστατεύσουν τα εθνικά συστήματα ασφαλείας, αρκεί να πιστοποιηθούν πρώτα από την NSA [44].

Ο AES, με μήκος κλειδιού 128-bit, έχει ήδη σπάσει, αφού χρειάστηκαν 2^{120} δοκιμές κλειδιών. Το 2005, ο D. J. Bernstein ανακοίνωσε το σπάσιμο ενός εξυπηρέτη που χρησιμοποιούσε κρυπτογράφηση AES. Η επίθεση έγινε με τη μέθοδο της χρονομέτρησης και χρειάστηκαν 200 εκατομμύρια επιλεγμένα κείμενα να χρονομετρηθούν για να βρεθεί το κλειδί. Παρόλα αυτά, μέσα στο διαδίκτυο τα πράγματα είναι αρκετά διαφορετικά και δεν είναι τόσο εύκολη μια τέτοιου είδους επίθεση.

Την ίδια χρονιά ο Shamir κατάφερε να ανακτήσει το μυστικό κλειδί του AES με την ίδια μέθοδο και μετά από 800 δοκιμές μέσα σε 65 milliseconds. Όμως, οι δοκιμές έγιναν στο ίδιο υπολογιστικό σύστημα που έτρεχε ο αλγόριθμος.

4.5 Συμπεράσματα.

Η συμμετρική κρυπτογραφία συμπεριλαμβάνει 2 συμμετέχοντες που μοιράζονται το ίδιο κλειδί. Επομένως, η συμμετρική κρυπτογραφία είναι ιδανική για ορισμένες εφαρμογές στις οποίες χρειάζεται να γίνει γρήγορη κρυπτογράφηση. Αντίθετα, στην περίπτωση όπου χρειάζεται η ανταλλαγή του συγκεκριμένου κλειδιού με τη δημιουργία κάποιου άμεσου καναλιού μεταξύ αυτών των δύο, η λύση της συμμετρική κρυπτογραφίας δεν προτείνεται. Τέτοια περίπτωση είναι αυτή του ηλεκτρονικού εμπορίου. Όμως παρόλα αυτά υπάρχουν περιπτώσεις όπου ή συμμετρική κρυπτογραφία αποτελεί μια πιο φθηνή λύση και προτιμάται. Συνοπτικά θα μπορούσαμε να καταγράψουμε τις ιδιότητες της συμμετρικής κρυπτογραφίας στον παρακάτω πίνακα [45]:

Λειτουργικότητα	Επαρκής επικοινωνία μεταξύ των συμμετεχόντων σε ένα κλειστό περιβάλλον
Υπολογιστική ικανότητα	Γρήγορη κρυπτογράφηση και αποκρυπτογράφηση με τη χρήση πιο απλών μαθηματικών
Μέγεθος κλειδιού	Συνήθως 128-bit, αλλά για μεγαλύτερη ασφάλεια γίνεται χρήση και 192-bit ή 256-bit
Υλικό	Απλό υλικό που χρησιμοποιούμε και στη χρήση απλών υπολογιστικών συστημάτων και όχι αρκετά ακριβό
Ασφάλεια	Η ασφάλεια εξαρτάται από το είδος της εφαρμογής, από το μήκος κλειδιού που θα χρησιμοποιήσουμε και τα μαθηματικά που χρησιμοποιεί ο αλγόριθμος

Από ότι είδαμε στα προηγούμενα κεφάλαια, ο DES έχει πια ξεπεραστεί και η χρήση του πια γίνεται μόνο για εκπαιδευτικούς λόγους. Το μήκος του κλειδιού που χρησιμοποιεί, το οποίο είναι 56-bit, χρειάζεται $2^{56} = 7 \cdot 10^{16}$ δοκιμές κλειδιών για να σπάσει. Με τη σημερινή τεχνολογία η δοκιμή αυτών των κλειδιών γίνεται μέσα σε ένα λογικό χρονικό διάστημα το οποίο δε καθιστά ασφαλή τον αλγόριθμο. Οι Biham και Shamir κατάφεραν να σπάσουν τον αλγόριθμο μόνο με 2^{47} δοκιμές αντί για 2^{56} και έτσι μείωσαν τον χρόνο σπασίματος του αλγορίθμου σε $1/500^{-ιοστο}$ του αρχικού χρόνου. Για να γίνει αυτό χρειάστηκαν 2^{47} ζευγάρια κειμένου/κρυπτοκειμένου [46].

Μετά τον DES δημιουργήθηκαν αρκετοί αλγόριθμοι που τον αντικατέστησαν, όπως ο 3-DES, ο DESX και ο IDEA, μέχρι που ο AES ήρθε να αντικαταστήσει τον DES και χρησιμοποιείται ακόμα παρέχοντας αρκετή ασφάλεια.

Ολοκληρώνοντας τα συμπεράσματα χρήσης των αλγορίθμων συμμετρικής κρυπτογράφησης στο παρακάτω πίνακα παρουσιάζεται σε ποιες περιπτώσεις συνίσταται η χρήση τους και σε ποιες όχι:

Είδος Εφαρμογής και Περίπτωση	**Ναι / Όχι**
Ηλεκτρονικό Εμπόριο	Όχι
Όταν έχουμε μετακίνηση κλειδιών	Όχι
Υπογραφές	Όχι
Ασφάλεια	Ναι / Όχι (Ανάλογα το κλειδί και τον αλγόριθμο)
Ταχύτητα Επεξεργασίας του αλγορίθμου	Ναι
Μήκους κλειδιού μικρό	Ναι
Κρυπτογράφηση σε ένα μόνο τερματικό	Ναι
Κρυπτογράφηση σε smart-cards	Ναι
Κρυπτογράφηση σε δίσκο (κυρίως USB)	Ναι
Οικονομία σε υλικό	Ναι

ΚΕΦΑΛΑΙΟ 5

5. Αλγόριθμοι Ασύμμετρης Κρυπτογραφίας.

5.1 Αλγόριθμος ανταλλαγής κλειδιών Diffie-Hellman.

5.1.1 Η Ιστορία του Αλγόριθμου.

Στην ασύμμετρη, αλλά κυρίως στην συμμετρική κρυπτογραφία, όπως έχουμε εξηγήσει προηγουμένως, απαιτείται η ασφαλής ανταλλαγή των κλειδιών μεταξύ των 2 συμμετεχόντων. Το πιο γνωστό πρωτόκολλο ανταλλαγής κλειδιών είναι αυτό του Diffie και Hellman (Diffie-Hellman key exchange protocol). Το πρωτόκολλο δημοσιεύτηκε το 1976 από τους Diffie W. και Hellman M., λίγο πριν την εμφάνιση του αλγορίθμου RSA που σήμανε και την έναρξη ουσιαστικά της κρυπτογραφίας δημοσίου κλειδιού. Παρόλα αυτά ο ίδιος ο Hellman αναφέρει το 2002 ότι ο Merkle R. πολύ νωρίτερα είχε ασχοληθεί με το θέμα αυτό και για αυτόν τον λόγο και ο αλγόριθμος θα έπρεπε να ονομαστεί "Diffie-Hellman-Merkle key exchange protocol" [47].

5.1.2 Περιγραφή του Αλγόριθμου.

Ας δούμε περιληπτικά πως γίνεται η ανταλλαγή κλειδιού. Σε επόμενα κεφάλαια θα γίνει εκτενέστερη αναφορά. Έστω, η Alice και ο Bob θέλουν να ανταλλάξουν κλειδί και να το ξέρουν μόνο αυτοί.

1. Συμφωνούν και διαλέγουν ένα πεπερασμένο σώμα αριθμών G και τον γεννήτορα g.
2. Η Alice επιλέγει έναν τυχαίο αριθμό a και στέλνει στον Bob τον g^a.
3. Ο Bob επιλέγει έναν τυχαίο αριθμό b και στέλνει στον Bob τον g^b.
4. Η Alice υπολογίζει το κλειδί $(g^b)^a$.
5. Ο Bob υπολογίζει το κλειδί $(g^a)^b$.

5.1.3 Παράδειγμα του Αλγόριθμου.

Έστω την ώρα που ο Bob στέλνει το κλειδί στην Alice, η Eve προσπαθεί να το κλέψει. Έστω ότι έχουμε [47]:

s = 2, το μυστικό κλειδί που πρέπει να μεταφερθεί κρυφά

a = 6, το ιδιωτικό κλειδί της Alice

b = 15, το ιδιωτικό κλειδί του Bob

g = 5, ο γεννήτορας (βάση) του σώματος

p = 23, ένας πρώτος αριθμός

Τότε σε κάθε φάση του αλγόριθμου θα ισχύει:

Alice		**Bob**		**Eve**	
γνωστό	άγνωστο	γνωστό	άγνωστο	γνωστό	άγνωστο
p = 23	b = 15	p = 23	a = 6	p = 23	a = 6
g = 5		g = 5		g = 5	b = 15
a = 6		b = 15			s = 2
$5^6 \bmod 23 = 8$		$5^{15} \bmod 23 = 19$		$5^a \bmod 23 = 8$	
$5^b \bmod 23 = 19$		$5^a \bmod 23 = 8$		$5^b \bmod 23 = 19$	
$19^6 \bmod 23 = 2$		$8^{15} \bmod 23 = 2$		$19^a \bmod 23 = s$	
$8^b \bmod 23 = 2$		$19^a \bmod 23 = 2$		$8^b \bmod 23 = s$	
$19^6 \bmod 23 = 8^b \bmod 23$		$8^{15} \bmod 23 = 19^a \bmod 23$		$19^a \bmod 23 = 8^b \bmod 23$	
s = 2		s = 2			

Έτσι, βλέπουμε ότι κατά την μετάδοση των δεδομένων το μόνο που μπορεί να κάνει η Eve είναι να ξέρει τα g, p, g^a mod p = 8 και g^b mod p = 19, αλλά όχι το s που είναι και το μυστικό κλειδί που κατάφερα η Alice να μεταδώσει στον Bob.

5.1.4 Ασφάλεια του Αλγόριθμου.

Η ασφάλεια του αλγορίθμου καταρρίπτεται μόνο αν ο υποκλοπέας ανακτήσει το g^{ab}. Αυτό όμως είναι αρκετά δύσκολο αφού τα G και g είναι πάρα πολύ μεγάλα και η εύρεση του διακριτού λογαρίθμου είναι δύσκολη διαδικασία. Παρόλα αυτά ο αλγόριθμος δεν παρέχει πιστοποίηση από μόνος του. Επομένως, αν κάποιος καταφέρει να αποκτήσει ταυτόχρονη επικοινωνία και με τους δυο συμμετέχοντες (Bob και Alice), κάνοντας να πιστέψουν ότι ο Bob είναι η Alice και αντίστροφα, θα μπορέσει να βρει το κλειδί. Για αυτό το λόγο, συνήθως ο αλγόριθμος λειτούργει παράλληλα με ένα σύστημα πιστοποίησης και όχι μόνος του.

5.2 Αλγόριθμος RSA (Rivest Shamir Adelman Algorithm).

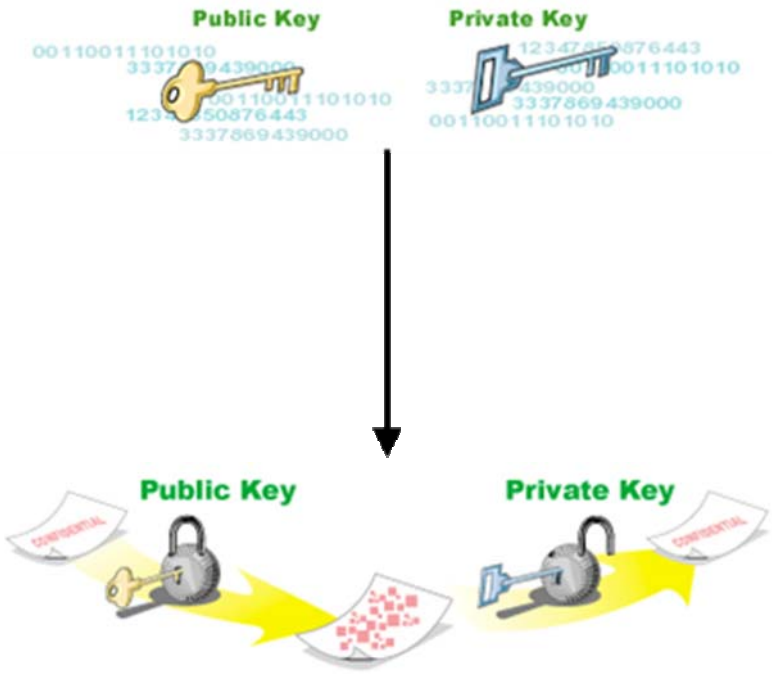
5.2.1 Η Ιστορία του Αλγόριθμου.

Το 1977, ένα χρόνο μετά τη δημοσίευση του πρωτοκόλλου του Diffie-Hellman, τρεις ερευνητές από το Massachusetts Institute of Technology (MIT), δημιούργησαν μια πρακτική μέθοδο χρησιμοποιώντας υπάρχουσες προτεινόμενες ιδέες. Αυτή η μέθοδος οδήγησε στη δημιουργία του αλγορίθμου RSA, που πήρε το όνομα του από τα αρχικά των τριών δημιουργών του, Rivest R., Shamir A. και Adelman L. [7].

Ο RSA αποτέλεσε πιθανόν τον πιο πολυχρησιμοποιημένο αλγόριθμο στην κρυπτογραφία δημοσίου κλειδιού. Επίσημα, χρησιμοποιήθηκε στις Η.Π.Α. το 1983, όταν και υιοθετήθηκε σαν επίσημος αλγόριθμος κρυπτογραφίας στη συγκεκριμένη χώρα. Ο αλγόριθμος θα χρησιμοποιούνταν επίσημα στη χώρα μέχρι το 2003, αλλά το 2000 εκδόθηκε επίσημα στις Η.Π.Α. η ανάλυση και η περιγραφή του και έτσι έγινε πια γνωστός σε όλους [49].

5.2.2 Η Περιγραφή του Αλγόριθμου.

Ο αλγόριθμος βασίζεται στη κρυπτογραφία δημοσίου κλειδιού. Το δημόσιο κλειδί δεν αποτελεί μυστική πληροφορία και έτσι μπορεί να μεταδοθεί χωρίς να απαιτείται η ύπαρξη κάποιου ασφαλούς μέσου μετάδοσης. Αντίθετα, το ιδιωτικό κλειδί δε μεταδίδεται ποτέ και χρησιμοποιείται μόνο από τον ιδιοκτήτη του.



Σχήμα 1: Λειτουργία κρυπτογραφίας δημόσιου κλειδιού.

Ο αλγόριθμος RSA λειτουργεί ως εξής:

1. Εύρεση ***μεγάλου*** ακεραίου, ως γινόμενο δύο ***πρώτων*** αριθμών.
2. Κατασκευή ***μυστικού - ιδιωτικού*** κλειδιού και ***δημόσιου*** κλειδιού (θεωρία αριθμών).
3. Όποιος επιθυμεί να στείλει κωδικοποιημένο μήνυμα, το κάνει με το ***δημόσιο*** κλειδί.
4. ***Μόνο*** όποιος έχει το αντίστοιχο ***μυστικό*** κλειδί μπορεί να αποκρυπτογραφήσει το μήνυμα.
5. ***Ανακάλυψη*** (δύσκολο!) των δύο πρώτων αριθμών ⇒ υποκλοπή μηνυμάτων εύκολη!

Ο πολλαπλασιασμός 2 ακεραίων είναι εύκολη υπόθεση:

$122 \cdot 831 = X \rightarrow X = 101382$

Όμως η εύρεση των δύο πρώτων αριθμών που αν πολλαπλασιαστούν μας δίνουν έναν γνωστό αριθμό είναι δύσκολη υπόθεση:

$415711 = X \cdot Y \rightarrow X=?$ και $Y=? \rightarrow X=199$ και $Y=2089$

Σε αυτό το πρόβλημα βασίζεται και ο RSA. Το περίφημο πρόβλημα της παραγοντοποίησης. Ας δούμε ένα **παράδειγμα (1)** για να καταλάβουμε πως γίνεται η κρυπτογράφηση:

Διαλέγω 2 πρώτους αριθμούς: p=11 και q=19.

Έτσι έχω: $n=11 \cdot 19=319$ και $\varphi(n)=(p-1)*(q-1)=(11-1)*(19-1)=10*18=180$.

Το δημόσιο κλειδί είναι το $k_e=101$ και υπολογίζω το ιδιωτικό κλειδί k_d:

$$k_d k_e \equiv 1(\mathrm{mod}\,\phi(n)) \Rightarrow k_d \cdot 101 \equiv 1(\mathrm{mod}\,180)$$

Η παραπάνω modular γραμμική εξίσωση έχει μόνο μία λύση αφού gcd(α,n)=gcd(101,180)=1.

Εφαρμόζοντας τον εκτεταμένο αλγόριθμο του Ευκλείδη2(α,n) (βλέπε παράρτημα Γ) για α=101, n=180 και έχοντας b=1 έχουμε:

$n=180=\alpha+79$	$79=n-\alpha$
$a=101=79+22$	$22=\alpha-(n-\alpha)=2\alpha-n$
$79=3 \cdot 22+13$	$13=n-\alpha-3(2\alpha-n)=-7\alpha+4n$
$22=13+9$	$9=2\alpha-n+7a-4n=9\alpha-5n$
$13=9+4$	$4=-7\alpha+4n-9\alpha+5n=-16\alpha+9n$
$9=2 \cdot 4+1$	$1=9\alpha-5n-2 \cdot (-16\alpha+9n)=9\alpha-5n+32\alpha-18n=41\alpha-23n$
$4=4 \cdot 1+0$	

Έτσι, στο τέλος έχουμε:

Ευκλείδης2(101,180) → (d,x',y')=(1,41,-23) και

$d = 101 \cdot 41 + 180 \cdot (-23) = 1$ με K_e=101 και K_d=41

Αφού έχουμε τα 2 κλειδιά έστω ότι θέλουμε να κρυπτογραφήσουμε τη λέξη [ρέμα]. Αντιστοιχούμε το Α→10, Β→11, Γ→12,.....Ω→33 έτσι ώστε όλα τα σύμβολα απλού κειμένου να κωδικοποιούνται με σταθερό πλήθος ψηφίων (2).

Έτσι το [ρέμα] θα γίνει → [26 14 21 10].

Κατόπιν, ομαδοποιούμε τα ψηφία ώστε οι αριθμοί που δημιουργούνται να είναι μικρότεροι του δημόσιου modulo 319 και έχουμε:

[ρέμα] → [261, 42, 110]

Μετά την κρυπτογραφική πράξη (εκτελείται 3 φορές) και θα έχουμε:

- 261^{101}=261 mod 319
- 42^{101}=196 mod 319
- 110^{101}=132 mod 319

Παράδειγμα κρυπτογράφησης με το RSA (2):

<u>Βήμα 1ο:</u> Επιλέγουμε δύο πρώτους αριθμούς p και q. Το γινόμενό τους είναι p·q=N. Τα p και q είναι γνωστά μόνο σε εμάς και άγνωστα για όλους τους άλλους. Αντιθέτως το N είναι γνωστό στο ευρύ κοινό.

Π.χ. Διαλέγω για p=3 και q=11 (πρώτοι)

Το γινόμενό τους είναι ίσο με: N=p·q=33 (αυτό είναι το δημόσιο κλειδί, το οποίο δημοσιεύεται είτε στο Internet, είτε σε έναν κατάλογο δημοσίων κλειδιών)

Τα p και q είναι άγνωστα στο ευρύ κοινό και αποτελούν το **ιδιωτικό κλειδί**.

<u>Βήμα 2ο:</u>

Υπολογίζω την συνάρτηση:

$\varphi(n)=(p-1) \cdot (q-1)$

Στο συγκεκριμένο παράδειγμα:

$\varphi(n)=(3-1) \cdot (11-1)=2 \cdot 10=20$

Στη συνέχεια επιλέγω έναν αριθμό που να μη δίνει υπόλοιπο μηδέν (σχετικά πρώτοι), όταν διαιρείται με το φ(n). Στο παράδειγμά μας το φ(n) είναι ίσο με 20.

Έστω ότι διαλέγω τον αριθμό 13,τον οποίο τον ονομάζω e και είναι το **κλειδί της κρυπτογράφησης**.

Βήμα 3ο:

Εφαρμόζω τον εκτεταμένο αλγόριθμο του Ευκλείδη με διαιρέτη τον e και διαιρετέο το φ(n). Ο Εκτεταμένος Αλγόριθμος του Ευκλείδη για 2 αριθμούς a, b που είναι σχετικά πρώτοι μεταξύ τους θα δώσει:

$\gcd(a, b)= a\cdot x + b\cdot y$

Επαλήθευση:$13\cdot (-3)+20\cdot 2=-39+40=1$

Στη συνέχεια υπολογίζω τον αριθμό d, ο οποίος είναι ίσος με $d=20+(-3)=17$

Ο αριθμός d είναι **το κλειδί αποκρυπτογράφησης** και θα χρησιμοποιηθεί αργότερα.

Βήμα 4ο:

Αντικαθιστώ κάθε γράμμα του αλφάβητου μ' έναν αριθμό:

A 10	B 11	C 12	D 13	E 14
F 15	G 16	H 17	I 18	J 19
K 20	L 21	M 22	N 23	O 24
P 25	Q 26	R 27	S 28	T 29
U 30	V 31	W 32		
X 33	Y 34	Z 35		

Έστω ότι θέλουμε να κρυπτογραφήσουμε τη αγγλική λέξη

JUPITER

Έχοντας αντιστοιχήσει κάθε γράμμα της αλφαβήτου με έναν αριθμό, η λέξη JUPITER αντιστοιχίζεται ως εξής:

J U P I T E R

ή

19 30 25 18 24 14 27

Στη συνέχεια ενώνω τους παραπάνω αριθμούς σ' ένα string. Οπότε θα προκύψει το εξής string: 19302518241427

Το παραπάνω string θα το χωρίσω σε μπλοκ, τα οποία αριθμητικά θα είναι μικρότερα του N (N=33). Δηλαδή το string θα χωριστεί ως εξής :

19-30-25-18-24-14-27

Στο κάθε μπλοκ εφαρμόζεται η συνάρτηση κρυπτογράφησης η οποία είναι η εξής: $E(b) = b^e \bmod n$

Άρα τα μπλοκ 19 30 25 18 24 14 27 που αντιστοιχούν στα γράμματα J-U-P-I-T-E-R θα μετατραπούν σύμφωνα με την παραπάνω συνάρτηση σε:

$19^{13} \bmod 33 = 28$

$30^{13} \bmod 33 = 6$

$25^{13} \bmod 33 = 16$

$18^{13} \bmod 33 = 24$

$29^{13} \bmod 33 = 2$

$14^{13} \bmod 33 = 5$

$27^{13} \bmod 33 = 15$

Τώρα για την αποκρυπτογράφηση, αυτή θα γίνει ως εξής:

$D(\alpha) = a^d \bmod n$

α: το κρυπτογραφημένο μπλοκ.

d: το κλειδί αποκρυπτογράφησης.

Έτσι θα έχουμε:

$$28^{17} \bmod 33 = 19$$
$$6^{17} \bmod 33 = 30$$
$$16^{17} \bmod 33 = 25$$
$$24^{17} \bmod 33 = 18$$
$$2^{17} \bmod 33 = 29$$
$$5^{17} \bmod 33 = 14$$
$$15^{17} \bmod 33 = 27$$

Η πιο κομψή πτυχή του κρυπτογράμματος RSA είναι ότι αν και το N είναι γνωστό, τα p και q είναι εξαιρετικά δύσκολο να βρεθούν, καθώς είναι δύσκολη η παραγοντοποίηση του N σε δύο πρώτους αριθμούς. Τα p και q είναι ένα ζευγάρι πρώτων αριθμών, οι οποίοι αποτελούνται από 100-200 ψηφία, ίσως και περισσότερα.

5.2.3 Η Ασφάλεια του Αλγόριθμου.

Η κρυπτογραφία δημόσιου κλειδιού προσφέρει αρκετά μεγάλη ασφάλεια. Έχει όμως το μειονέκτημα ότι οι αλγόριθμοι δημοσίου κλειδιού είναι πολύ πιο αργοί από τους αλγόριθμους συμμετρικής κρυπτογράφησης. Το σχήμα του RSA βασίζεται στο ότι αν και το N είναι γνωστό, τα p και q είναι εξαιρετικά δύσκολο να βρεθούν, καθώς είναι δύσκολη η παραγοντοποίηση του N σε δύο πρώτους αριθμούς. Το γεγονός αυτό καθιστά το κρυπτόγραμμα RSA δύσκολο και αρκετά χρονοβόρο στο να αποκρυπτογραφηθεί.

Το μεγάλο πλεονέκτημα της κρυπτογραφίας δημόσιου κλειδιού που εφαρμόζει ο RSA είναι ότι καταργεί όλα τα προβλήματα που συνδεόταν με τα παραδοσιακά κρυπτογράμματα και τις μεθόδους ανταλλαγής κλειδιών. Όμως, ο RSA είναι πολύ πιο αργός από τον DES και άλλα συμμετρικά κρυπτοσυστήματα [7].

Ο Kocher περιέγραψε ένα νέο είδος επίθεσης στον RSA το 1995, όπου αν ο επιτιθέμενος γνωρίζει το υλικό, με απόλυτη ακρίβεια, με το οποίο κάποιος κάνει την κρυπτογράφηση μπορεί να μετρήσει τον χρόνο που κάνει το μηχάνημα να κρυπτογραφήσει διάφορα κρυπτοκείμενα. Έτσι, υπολογίζει το ιδιωτικό κλειδί. Μια

λύση σε αυτού σε αυτές τις επιθέσεις είναι η χρήση ίσης ποσότητας χρόνου κρυπτογράφησης για όλα τα κρυπτοκείμενα ανεξάρτητα του μεγέθους τους. Όμως, έτσι μειώνεται η απόδοση του αλγορίθμου. Ακόμα καλύτερα είναι η χρήση της τεχνικής Blinding για τον αποπροσανατολισμό και τη δημιουργία ενός random αλγόριθμου του χρόνου κρυπτογράφησης των μηνυμάτων.

Η ασφάλεια που προσφέρει ένα κλειδί μεγέθους 1024-bit είναι αρκετά καλή. Η RSA Laboratories προτείνει τη χρήση κλειδιών 2048-bit. Για συνηθισμένη χρήση, συνήθως προτείνεται κλειδί μήκους 768-bit, το οποίος όμως σπάει εύκολα με διάφορες τεχνικές. Η RSA Laboratories το 1995 ανέφερε ότι ένα κλειδί μήκους 512-bit μπορεί να σπάσει σε 8 μήνες με κόστος 1 εκατομμύριο δολάρια. Και πράγματι, το 1999, σε διαγωνισμό που προκήρυξε η RSA ένα κλειδί 512-bit έσπασε σε 7 μήνες. Επίσης, πρέπει να σημειωθεί ότι διπλασιάζοντας τον αριθμό modulo τετραπλασιάζεται ο χρόνος κρυπτογράφησης, το οποίο είναι εντελώς ασύμφορο.

5.3 Οι Συναρτήσεις Hash.

5.3.1 Τι είναι το hashing.

Μια πολύ πρακτική μέθοδος κρυπτογράφησης είναι η συνάρτηση κατακερματισμού ή κατατεμαχισμού (one way hash). Το hashing στην πληροφορική έχει την έννοια της αυθαίρετης – τυχαίας ανακατάταξης μιας σειράς από δεδομένα με αποτέλεσμα μια σταθερή σειρά από δεδομένα. Το αρχικό μήκος μπορεί να διαφέρει αλλά το αποτέλεσμα θα είναι πάντα ίδιου μήκους, συνήθως 128 ή 160 bits στην κρυπτογραφία.

Το hashing χρησιμοποιείται ευρέως στη δημιουργία ευρετηρίων για τη γρήγορη αναζήτηση. Η τεχνική λειτουργίας τους δε διαφέρει και πολύ από τη χρήση τους στη κρυπτογραφία. Για κάθε μήνυμα δημιουργείται μια σύνοψή του, που είναι μια σειρά από bits με συγκεκριμένο πλήθος, για παράδειγμα 128 bits. Η σύνοψη του μηνύματος, που είναι γνωστή με τον όρο fingerprint ή message digest, αποτελεί ψηφιακή αναπαράσταση του μηνύματος και είναι μοναδική για το μήνυμα που αντιπροσωπεύει.

Το πρόβλημα με το hashing είναι ότι πολλά διαφορετικά κείμενα μπορεί να μετατραπούν σε όμοιες σειρές από string. Έτσι, στη περίπτωση των hash συναρτήσεων που επιστρέφουν 128-bit string οι τιμές της επιστρεφόμενης τιμής είναι $3{,}4 \cdot 10^{38}$. Όμως, τα κείμενα που κρυπτογραφούνται είναι άπειρα. Έτσι, διαφορετικά κείμενα μπορεί να επιστρέψουν ίδιες τιμές. Παρόλα αυτά πιθανότητες να γίνει κάτι τέτοιο έχει ελαχιστοποιηθεί σχεδόν στο μηδέν με τους καινούργιους αλγόριθμους hash. Γνωστές είναι και οι επιθέσεις, οι λεγόμενες "birthday attack", όπου ο επιτιθέμενος καταφέρνει να δημιουργήσει ένα μήνυμα που δίνει την ίδια σύνοψη μηνύματος με το αυθεντικό μήνυμα για να παραπλανήσει τον παραλήπτη του μηνύματος.

Αν αλλάξουμε έστω και μια τελεία στο μήνυμα, θα αλλάξει και η σύνοψή του, ενώ είναι πρακτικά αδύνατο δύο διαφορετικά μηνύματα να δώσουν την ίδια σύνοψη. Είναι επίσης πρακτικά αδύνατο να ανακτήσουμε το αρχικό μήνυμα αν γνωρίζουμε τη σύνοψή του.

5.3.2 Πως λειτουργεί το hashing.

Η λειτουργία του hashing γίνεται ως εξής:

1. Ο αποστολέας δημιουργεί τη σύνοψη του μηνύματος (message digest) που θέλει να αποστείλει, χρησιμοποιώντας κάποιον αλγόριθμο κατακερματισμού και έτσι δημιουργείται μια σειρά από bits με συγκεκριμένο μήκος.
2. Ο αποστολέας κρυπτογραφεί μετά με το ιδιωτικό του κλειδί τη σύνοψη που έχει δημιουργηθεί.
3. Η κρυπτογραφημένη σύνοψη προσαρτάται στο αρχικό κείμενο και μεταδίδεται μαζί του, χωρίς να είναι απαραίτητη και η κρυπτογράφηση του αρχικού κειμένου.
4. Στη λήψη, ο παραλήπτης αποσπά από το μήνυμα την κρυπτογραφημένη σύνοψη και εφαρμόζει στο κανονικό κείμενο τον ίδιο αλγόριθμο κατακερματισμού, ώστε να δημιουργήσει μια δική του σύνοψη.

5. Μετά αποκρυπτογραφεί με το δημόσιο κλειδί του αποστολέα την κρυπτογραφημένη σύνοψη του μηνύματος και συγκρίνει τις δύο συνόψεις.
6. Αν οι δύο αυτές συνόψεις βρεθούν ίδιες, αυτό σημαίνει ότι το μήνυμα που έλαβε ο παραλήπτης είναι ακέραιο, ενώ αν βρεθούν διαφορετικές, θα σημαίνει ότι το μήνυμα αλλοιώθηκε κατά τη μετάδοσή του.

Μία hash συνάρτηση λειτουργεί προς μία κατεύθυνση και έτσι δε γίνεται να πάρουμε το κείμενο από το κρυπτοκείμενο με την αντίθετη λειτουργία.

5.3.3 SHA (Secure Hash Algorithm).

Ο Secure Hash Algorithm (SHA) αναπτύχθηκε από το NIST και χρησιμοποιήθηκε το 1994 από πολλά κρυπτογραφικά πρωτόκολλα, με την ονομασία SHA-1. Στη πορεία βγήκαν και άλλες παραλλαγές του αλγορίθμου όπως, SHA-2, SHA-224, SHA-256, SHA-384 και SHA-512. Το SHA-3 είναι υπό κατασκευή και θα είναι έτοιμο το 2012 [52]. Ο SHA-1 δημιουργεί συνόψεις των 160 bits και μπορεί να δεχτεί στην είσοδο μέχρι 264 bits μήκος κειμένου. Ο ψευδοκώδικας και ένα παράδειγμα του SHA-1 παρουσιάζεται στο παράρτημα Δ.

Η λειτουργία της SHA-1 περιγράφεται στα παρακάτω βήματα [53]:

- Βήμα 1 → Αρχική συμπλήρωση του μηνύματος.
- Βήμα 2 → Προσάρτηση του αρχικού μήκους.
- Βήμα 3 → Αρχικοποίηση της προσωρινής μνήμης υπολογισμών.
- Βήμα 4 → Εκκίνηση των υπολογισμών σε 4 στάδια.
- Βήμα 5 → Έξοδος της συγχώνευσης μηνύματος μήκους 160-bit.

Αρχική συμπλήρωση του μηνύματος:

Το αρχικό μήνυμα συμπληρώνεται με επιπλέον bits με τέτοιο τρόπο ώστε το μήκος του να είναι 448-bit με υπόλοιπο ως προς το 512. Με άλλα λόγια το αρχικό μήνυμα συμπληρώνεται ώστε το μήκος του να είναι 64-bit λιγότερο από το 512. Η αρχική συμπλήρωση γίνεται ακόμα και αν το μήνυμα έχει το επιθυμητό μήκος. Π.χ. εάν το αρχικό μήνυμα έχει μήκος 448-bits τότε συμπληρώνεται από άλλα 512-bits ώστε να φθάσει αθροιστικά τα 960-bit. Συνεπώς ο αριθμός των bits συμπλήρωσης

είναι από 1 έως 512. Το συμπλήρωμα έχει τη μορφή 1000…..000, δηλαδή ένα απλό 1-bit ακολουθούμενο από όσα 0-bit χρειάζονται.

Προσάρτηση του αρχικού μήκους:

Το αρχικό μήκος του μηνύματος προσαρτάται στο τέλος του μηνύματος ως ένας 64-bit ακέραιος. Μία προσωρινή μνήμη 160-bit χρησιμοποιείται για να διαχειριστεί τα ενδιάμεσα και τα τελικά αποτελέσματα της SHA-1. Η προσωρινή μνήμη αναπαρίσταται σαν 5 καταχωρητές των 32-bit, A, B, C, D και E.

Αρχικοποίηση της προσωρινής μνήμης υπολογισμών:

Οι καταχωρητές αρχικοποιούνται ως εξής:

$A = 67452301_{(16)}$

$B = EFCDAB89_{(16)}$

$C = 98BADCFE_{(16)}$

$D = 10325476_{(16)}$

$E = C3D2E1F0_{(16)}$

Εκκίνηση των υπολογισμών σε 4 στάδια:

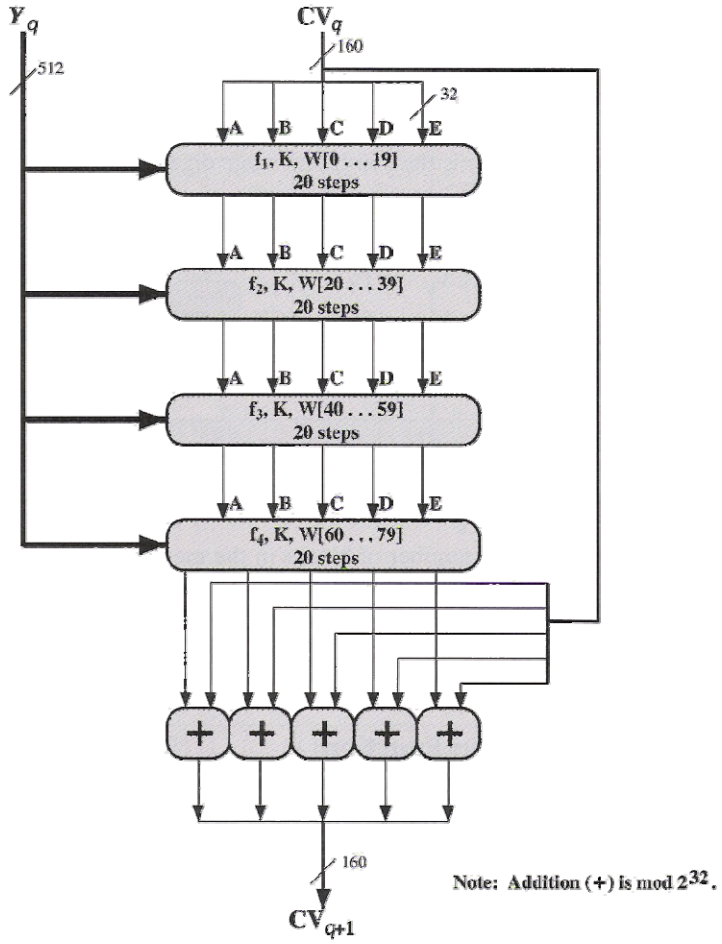
Η SHA-1 έχει 4 στάδια υλοποίησης. Κάθε στάδιο αποτελείται από 20 βήματα. Κάθε στάδιο έχει παρόμοια δομή αλλά χρησιμοποιεί διαφορετική λογική συνάρτηση και για κάθε στάδιο η λογική συνάρτηση αναφέρεται ως f_1, f_2, f_3 και f_4.

Σε κάθε στάδιο εισάγεται το 512-bit τμήμα Y_q και η τιμή της 160-bit ενδιάμεσης μνήμης ABCDE. Κάθε στάδιο κάνει χρήση της προσθετικής σταθεράς K_t, όπου $0 \leq t \leq 79$, για κάθε ένα από τα 80 βήματα. Συγκεκριμένα:

Step Number	Hexadecimal	Take Integer Part of:
$0 \leq t \leq 19$	K_t = 5A827999	$[2^{30} \times \sqrt{2}]$
$20 \leq t \leq 39$	K_t = 6ED9EBA1	$[2^{30} \times \sqrt{3}]$
$40 \leq t \leq 59$	K_t = 8F1BBCDC	$[2^{30} \times \sqrt{5}]$
$60 \leq t \leq 79$	K_t = CA62C1D6	$[2^{30} \times \sqrt{10}]$

Η έξοδος από το 4ο στάδιο για το τμήμα CV_q είναι η είσοδος για το 1ο στάδιο για το τμήμα CV_{q+1}. Όλες οι προσθέσεις γίνονται με mod 2^{32} ώστε η έξοδος

από κάθε τμήμα CV να είναι 160-bit. Αφού ολοκληρωθούν όλα τα 512-bit τμήματα παράγεται η 160-bit έξοδος από το L στάδιο (βλέπε σχήμα 2).



Σχήμα 2: Υλοποίηση του SHA-1.

5.4 Αλγόριθμος ElGamal.

5.4.1 Οι διακριτοί λογάριθμοι και οι κυριότερες εφαρμογές των διακριτών λογαρίθμων.

Ο κύριος λόγος για τον οποίο οι διακριτοί λογάριθμοι χρησιμοποιούνται τόσο πολύ στην κρυπτογραφία είναι γιατί έχουν τις ιδιότητες μιας συνάρτησης μοναδικής κατεύθυνσης (one-way). One-way συνάρτηση, y = f(x), είναι μία συνάρτηση για την

οποία ο υπολογισμός του y δοθέντος του x είναι πολύ εύκολος, ενώ αντίθετα, είναι πολύ δύσκολο να υπολογιστεί το x με δεδομένη την τιμή του y. Πράγματι και για τον διακριτό λογάριθμο, είναι πολύ εύκολο να υπολογιστεί το $y = g^x$ από τα g και x, ενώ είναι πολύ δύσκολο να υπολογιστεί το $\log_g y$. Η δυσκολία επίλυσης του DLP είναι τόσο μεγάλη έτσι ώστε για τα σώματα F_q με τάξη $q \approx 10^{150}$ και μεγαλύτερη, το πρόβλημα να είναι μη επιλύσιμο πρακτικά και τα σώματα αυτά να θεωρούνται επαρκώς ασφαλή για όλες τις κρυπτογραφικές εφαρμογές [12].

Η πιο απλή εφαρμογή του DLP αφορά την απόδειξη της γνησιότητας του χρήστη (user authentication). Για παράδειγμα σε ένα σύστημα πολλών χρηστών, κάθε χρήστης έχει ένα login και ένα password το οποίο πρέπει να μείνει κρυφό για οποιονδήποτε άλλο χρήστη. Το να καταχωρηθούν όλα τα passwords σε ένα αρχείο είναι αρκετά ριψοκίνδυνο γιατί κάποιος τρίτος θα μπορούσε να αποκτήσει πρόσβαση στο αρχείο και να εισβάλει στο σύστημα. Αντί λοιπόν να αποθηκεύεται ένα password x_i για κάθε χρήστη i, θα μπορούσε να επιλέγεται ένα πεπερασμένο σώμα F_q και ένα στοιχείο g και στο αρχείο να αποθηκεύονται οι τιμές $g^{x_i} = y_i \in F_q$ για κάθε χρήστη i και να συγκρίνεται το αποτέλεσμα με κάποιο άλλο αρχείο. Έτσι, ακόμα και αν κάποιος αποκτήσει πρόσβαση στο αρχείο, θα πρέπει πρώτα να λύσει το DLP για να βρει τα x_i [13].

Μια ακόμα εφαρμογή βρίσκεται στην ανταλλαγή κλειδιών μεταξύ δύο χρηστών. Δύο χρήστες που θέλουν να επικοινωνήσουν μυστικά σε ένα κανάλι που δεν είναι ασφαλές πρέπει πρώτα να συμφωνήσουν σε ένα μυστικό κλειδί κοινό και στους δύο, το οποίο και θα χρησιμοποιούν για να κωδικοποιούν και να αποκωδικοποιούν τα μηνύματα που στέλνει ο ένας στον άλλο. Ένας ασφαλής αλγόριθμος για ανταλλαγή κλειδιών είναι ο αλγόριθμος των Diffie και Hellman (Diffie-Hellman key exchange protocol). Σύμφωνα με τον αλγόριθμο αυτό, επιλέγεται ένα πεπερασμένο σώμα F_q και ένα στοιχείο g τα οποία είναι γνωστά και στους δύο χρήστες. Κάθε χρήστης επιλέγει ένα ιδιωτικό κλειδί x_i με $2 \le x_i \le q-2$, το οποίο είναι μυστικό σε όλους τους άλλους χρήστες και κάνει γνωστό το δημόσιο κλειδί του $g^{x_i} = y_i \in F_q$. Ο χρήστης A στέλνει το δημόσιο κλειδί του y_A στον B και ο B το δημόσιο κλειδί του y_B στον A. Έτσι τελικά και οι δύο μπορούν να μοιράζονται το ίδιο κλειδί $g^{x_A x_B} = y_A^{x_B} = y_B^{x_A}$ υψώνοντας ο καθένας το δημόσιο

κλειδί του άλλου σε δύναμη ίση με το ιδιωτικό τους κλειδί. Γίνεται κατανοητό λοιπόν ότι μόνο αν μπορέσει κάποιος να λύσει το DLP θα μπορούσε να σπάσει το σύστημα και να βρει το κοινό κλειδί $g^{x_A x_B}$ των δύο χρηστών.

Οι διακριτοί λογάριθμοι χρησιμοποιούνται επίσης στην κρυπτογράφηση και αποκρυπτογράφηση μηνυμάτων. Έστω ότι ο χρήστης Α επιθυμεί να στείλει ένα μήνυμα m στον χρήστη Β. Τότε επιλέγει ένα c, με $1 \le c \le q-1$ και $\gcd(c,\ q-1)=1$ (δηλαδή τα c και q - 1 είναι μεταξύ τους πρώτοι ή αλλιώς gcd(c, q- 1) = 1) και μεταδίδει το κρυπτογραφημένο μήνυμα $x = m^c$ στον Β. Ο Β με τη σειρά του επιλέγει έναν ακέραιο d, όπου $1 \le d \le q-1$ και gcd(d, q – 1) = 1 και μεταδίδει πίσω στον Α το μήνυμα $y = x^d = m^{cd}$. Ο Α υπολογίζει το $z = y^{c'}$ όπου $c \cdot c' = 1 (\mathrm{mod}\, q-1)$ και στέλνει το z στον Β. Ο Β τελικά, για να δει το αρχικό μήνυμα m, υπολογίζει το $z^{d'}$ με $d \cdot d' = 1 (\mathrm{mod}\, q-1)$ αφού $z^{d'} = m^{d \cdot d'} = m$. Επομένως κάποιος τρίτος μόνο αν έλυνε το DLP θα μπορούσε να ανακτήσει το μήνυμα m από τα μεταδιδόμενα κρυπτογραφημένα μηνύματα m^c, m^{cd} και m^d [13].

Τέλος, μία εφαρμογή του DLP βρίσκεται σε έναν αλγόριθμο για δημιουργία ψηφιακών υπογραφών. Οι ψηφιακές υπογραφές ενσωματώνονται στα μηνύματα που στέλνει για παράδειγμα ένας χρήστης Α, για να επιβεβαιώσουν τον χρήστη Β πως πράγματι τα μηνύματα στέλνονται από τον Α και όχι από κάποιον που προσποιείται ότι είναι ο Α. Έτσι, οι ψηφιακές υπογραφές αποτελούν δικλείδα ασφαλείας για τον Α ο οποίος μπορεί να αποδείξει πως ένα μήνυμα δεν στάλθηκε από αυτόν αλλά από κάποιον «πλαστογράφο» (π.χ. σε ένα δικαστήριο όπου ο Α μπορεί να αποδείξει έτσι την αθωότητά του).

5.4.2 Ιστορία του Αλγόριθμου του ElGamal.

Ο Taher ElGamal είναι ο εφευρέτης του ομώνυμου κρυπτοσυστήματος. Θεωρείται ως μια από τις πιο διακεκριμένες μορφές στην κρυπτογραφία και στη βιομηχανία της ασφάλειας πληροφοριών. Διετέλεσε ως βασικό επιστημονικό στέλεχος της Netscape Communications, όπου αποδείχτηκε πρωτοπόρος στις τεχνολογίες ασφάλειας του Διαδικτύου, όπως το SSL, που αποτελεί το πρότυπο για την ασφάλεια στο Internet. Ανέπτυξε επίσης έναν αριθμό από τρόπους πληρωμής

μέσω Internet. Επίσης, ο ElGamal διετέλεσε επίσης διευθυντής στην εταιρεία RSA Data Security, όπου παρήγαγε τα εργαλεία κρυπτογράφησης με τον RSA.

Η καριέρα του στην κρυπτογραφία ξεκίνησε με ένα διδακτορικό στο Stanford υπό την επίβλεψη του Martin Hellman και του Whitfield Diffie, οι οποίοι και έθεσαν τις βάσεις της κρυπτογραφίας δημόσιου κλειδιού. Το 1985 εφηύρε το ομώνυμο κρυπτοσύστημα, ενώ το 1994 η κυβέρνηση των ΗΠΑ υιοθέτησε το πρότυπο ψηφιακής υπογραφής (Digital Signature Standard, DSS), το οποίο βασίζεται στο κρυπτοσύστημα του ElGamal.

5.4.3 Περιγραφή του Αλγόριθμου του ElGamal.

Ο πλέον γνωστός αλγόριθμος δημιουργίας ψηφιακών υπογραφών είναι ο αλγόριθμος του ElGamal. Κάθε χρήστης A για να υπογράψει ένα μήνυμα m, με $1 \le m \le q-1$, υπολογίζει και στέλνει στους χρήστες με τους οποίους θέλει να επικοινωνήσει ένα ζευγάρι ακεραίων (r, s) με $1 \le r, s \le q-1$.

Τα r και s είναι ίσα με:

$$r \equiv g^k \bmod p$$

$$s \equiv k^{-1}(m - x_A r)$$

όπου ο k είναι ένας τυχαίος ακέραιος $1 \le k \le q-2$ τέτοιος ώστε $\gcd(k, q-1) = 1$. Το x_A είναι το ιδιωτικό κλειδί του A. Η υπογραφή του A επικυρώνεται από έναν χρήστη B, ο οποίος γνωρίζει το μήνυμα m, ως εξής: υπολογίζει δύο ποσότητες g^m και $(y_A)^r r^s \pmod q$ και αν τις βρει ίσες τότε πράγματι το μήνυμα m έχει υπογραφεί από τον A. Διαφορετικά κάποιος τρίτος έχει αλλοιώσει το μήνυμα αφού κάθε αλλαγή στο μήνυμα είναι καταστροφική για την υπογραφή. Οι δύο αυτές ποσότητες πρέπει να βγουν ίσες γιατί:

$$g^m = g^{x_A r + ks} = (g^{x_A})^r (g^k)^s = (y_A)^r r^s .$$

Το κλειδί x_A είναι απαραίτητο για την υπογραφή του μηνύματος και κάποιος τρίτος για να προσποιηθεί τον A θα πρέπει να το γνωρίζει. Αυτό όμως μπορεί να συμβεί μόνο αν μπορέσει και λύσει το DLP [13].

5.4.4 Ιδιότητες του Αλγόριθμου του ElGamal.

Ο ElGamal δημιουργεί κρυπτογραφημένο κείμενο σχεδόν 2 φορές το αρχικό κείμενο. Επομένως, απαιτεί μεγάλες αποθηκεύτηκες ικανότητες και επίσης μεγαλύτερο χρόνο για την αποκρυπτογράφηση του κειμένου. Είναι πιο αργός από τον RSA και φυσικά από τους DES και AES.

Η ασφάλεια του αλγορίθμου βασίζεται στο σώμα G που θα επιλέξουμε. Όσο μεγαλύτερο είναι τόσο μεγαλύτερη είναι και η ασφάλεια του κρυπτοσυστήματος. Η επίλυση του βλήματος DLP είναι πολύ πιο δύσκολη από την παραγοντοποίηση ακεραίων και έτσι δίνει μεγαλύτερη ασφάλεια από τον RSA.

5.5 Συμπεράσματα.

Συνοπτικά θα μπορούσαμε να πούμε ότι τα πλεονεκτήματα και τα μειονεκτήματα των αλγορίθμων της ασύμμετρης κρυπτογραφίας είναι τα εξής [45]:

Λειτουργικότητα	Χρησιμοποιείται περισσότερο στη κρυπτογραφία για την επικοινωνία μεταξύ απομακρυσμένων υπολογιστών και είναι αρκετά εύκολη η υλοποίησή της στις περισσότερες περιπτώσεις.
Υπολογιστική ικανότητα	Αργή κρυπτογράφηση με πολλούς μαθηματικούς υπολογισμούς και πιο πολύπλοκες λειτουργίες, οι οποίες βασίζονται σε δύσκολα μαθηματικά.
Μέγεθος κλειδιού	Κλειδί μεγέθους 1000-bit και άνω παρέχουν πολύ καλή ασφάλεια
Υλικό	Δημιουργία σύνθετων κυκλωμάτων για την υλοποίηση χρονοβόρων αλγορίθμων και επομένως απαίτηση πιο ισχυρών επεξεργαστών

Ασφάλεια	Η ασφάλεια εξαρτάται από το είδος της εφαρμογής, από το μήκος κλειδιού που θα χρησιμοποιήσουμε και τα μαθηματικά που χρησιμοποιεί ο αλγόριθμος

Οι αλγόριθμοι της συμμετρικής κρυπτογραφίας δεν καταναλώνουν τόση υπολογιστική ισχύ όσο καταναλώνουν οι αλγόριθμοι ασύμμετρης κρυπτογραφίας. Παρόλα αυτά τα τελευταία χρόνια αναπτύχθηκε μια νέα τεχνική κρυπτογραφίας που χρησιμοποιεί την θεωρία των ελλειπτικών καμπυλών. Με τη τεχνική αυτή με τη χρήση ίδιου μήκους κλειδιού παρέχεται πολύ μεγαλύτερη ασφάλεια σε σχέση με την ασφάλεια που παρέχει η ασύμμετρη και η συμμετρική κρυπτογραφία. Για αυτόν τον λόγο κρίνεται απαραίτητη και η ανάλυση της θεωρίας και χρήσης των ελλειπτικών καμπύλων στην κρυπτογραφία στο επόμενο κεφάλαιο.

ΚΕΦΑΛΑΙΟ 6

6. Κρυπτογραφία με Ελλειπτικές Καμπύλες

6.1 Εισαγωγή στη θεωρία των ελλειπτικών καμπυλών.

Το κεφάλαιο αυτό παρέχει σφαιρική εισαγωγή στις Ελλειπτικές Καμπύλες και στον τρόπο που αυτές χρησιμοποιούνται για τη δημιουργία ασφαλών και αποτελεσματικών κρυπτοσυστημάτων.

Στην αρχή θα μελετηθούν οι ιδιότητες των ελλειπτικών καμπυλών. Βέβαια κάποιο μαθηματικό υπόβαθρο, σχετικό με τη θεωρία της άλγεβρας, απαιτείται. Τα βασικότερα όμως εξηγήθηκαν σε προηγούμενο κεφάλαιο. Επίσης θα εξηγηθεί ο λόγος που κάνει τις ομάδες ελλειπτικών καμπυλών κατάλληλες για την κρυπτογραφία μέσα από την παρουσίαση του Προβλήματος του Διακριτού Λογαρίθμου στις Ελλειπτικές Καμπύλες (Elliptic Curve Discrete Logarithm Problem, ECDLP). Τέλος, θα εξηγηθεί πως οι ελλειπτικές καμπύλες και το ECDLP εφαρμόζονται στην ιδέα της κρυπτογραφίας.

Οι ελλειπτικές καμπύλες μελετήθηκαν εντατικά τα τελευταία 150 χρόνια, σαν αλγεβρικές - γεωμετρικές οντότητες, και από αυτές τις μελέτες προέκυψε μια πλούσια και μυστηριώδης θεωρία. Τα συστήματα ελλειπτικών καμπυλών εφαρμοσμένα στη κρυπτογραφία προτάθηκαν για πρώτη φορά το 1985 παράλληλα από τον Neal Koblitz (πανεπιστήμιο της Ουάσιγκτον) και από τον Victor Miller, που εργαζόταν για την IBM Yorktown Heights [10].

Πολλά κρυπτοσυστήματα συχνά απαιτούν τη χρήση αλγεβρικών ομάδων. Οι ελλειπτικές καμπύλες μπορούν να χρησιμοποιηθούν για το σχηματισμό ομάδων ελλειπτικών καμπυλών. Όπως εξηγήσαμε σε προηγούμενο κεφάλαιο μια ομάδα είναι ένα σετ από στοιχεία στα οποία ισχύουν καθορισμένες μαθηματικές πράξεις μεταξύ αυτών των στοιχείων. Για τις ομάδες ελλειπτικών καμπυλών, αυτές οι συγκεκριμένες πράξεις καθορίζονται γεωμετρικά. Εισάγοντας περισσότερους περιορισμούς στις ιδιότητες των στοιχείων μιας ομάδας, όπως για παράδειγμα περιορίζοντας τον

αριθμό των σημείων μιας καμπύλης, δημιουργείται μια γενεσιουργός περιοχή στοιχείων ομάδας ελλειπτικών καμπυλών.

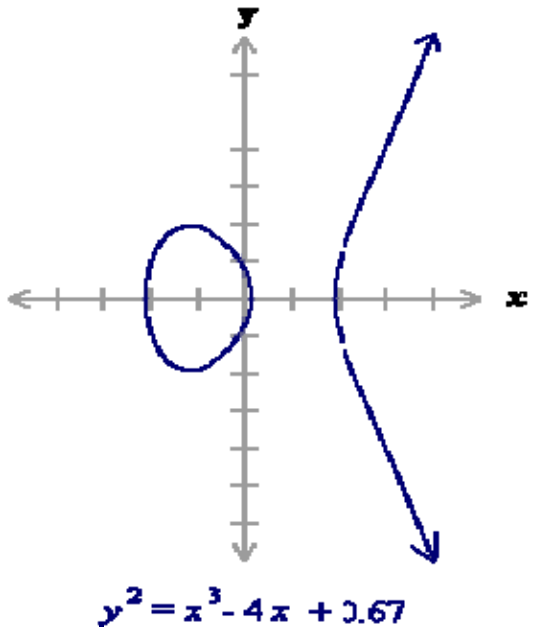
Στην εργασία αυτή, οι ελλειπτικές καμπύλες αρχικά εξετάζονται πάνω στους πραγματικούς αριθμούς έτσι ώστε να διευκρινιστούν οι γεωμετρικές ιδιότητες των ομάδων ελλειπτικών καμπυλών. Κατόπιν, οι ομάδες ελλειπτικών καμπυλών εξετάζονται στα υποκείμενα σώματα F_p (όπου το p είναι ένας πρώτος αριθμός) και F_{2^m} (μια δυαδική αναπαράσταση με 2^m στοιχεία).

6.2 Ομάδες Ελλειπτικών Καμπυλών με πραγματικούς αριθμούς.

Μια ελλειπτική καμπύλη με πραγματικούς αριθμούς ορίζεται από το σύνολο των σημείων (x, y) τα οποία ικανοποιούν την εξίσωση ελλειπτικής καμπύλης της μορφής:

$y^2 = x^3 + ax + b$, όπου x, y, a και b είναι πραγματικοί αριθμοί.

Κάθε επιλογή των αριθμών a και b παράγει και διαφορετική ελλειπτική καμπύλη. Για παράδειγμα, τα σημεία a = -4 και b = 0,67 δίνουν την ελλειπτική καμπύλη με εξίσωση $y^2 = x^3 - 4x + 0{,}67$. Το διάγραμμα αυτής της καμπύλης παρουσιάζεται παρακάτω:



Σχήμα 1: Παράδειγμα ελλειπτικής καμπύλης.

Για κάποιον συνδυασμό των a και b, η εξίσωση της ελλειπτικής καμπύλης δεν έχει πολλαπλές διαφορετικές ρίζες (για y = 0). Αν η $x^3 + ax + b$ περιέχει μη συνεχόμενους παράγοντες, ή ισοδύναμα αν το $4a^3 + 27b^2$ είναι διάφορο του 0, τότε η ελλειπτική καμπύλη $y^2 = x^3 + ax + b$ μπορεί να χρησιμοποιηθεί για το σχηματισμό μιας ομάδας. Μια ομάδα ελλειπτικής καμπύλης με πραγματικούς αριθμούς αποτελείται από σημεία της αντίστοιχης ελλειπτικής καμπύλης, μαζί με το ειδικό σημείο Ο που ονομάζεται σημείο στο άπειρο [8].

Τα σημεία μιας ελλειπτικής καμπύλης σχηματίζουν μια αβελιανή ομάδα, με πράξη την πρόσθεση. Το πώς γίνεται αυτό θα φανεί στη συνέχεια. Επίσης, πρέπει να σημειώσουμε ότι οι ελλειπτικές καμπύλες έκαναν την αρχική τους εμφάνιση κατά την διάρκεια προσπαθειών για τον υπολογισμό του μήκους τόξου σε μια έλλειψη. Για τον λόγο αυτό, αν και δεν έχουν καμία σχέση με τις ελλείψεις, πήραν το όνομα τους από αυτές [11].

Για να οριστεί μια ελλειπτική καμπύλη, όπως φαίνεται και από τον παραπάνω ορισμό, είναι απαραίτητος ο προσδιορισμός του μεγέθους του σώματος F, της αναπαράστασης του σώματος (π.χ. F_{2^m} ή F_p) και ασφαλώς των παραμέτρων a και b. Συνήθως, μαζί με όλες αυτές τις παραμέτρους ορίζεται και ένα σημείο G που είναι ο γεννήτορας των άλλων σημείων της ελλειπτικής καμπύλης. Δηλαδή αυτό είναι ένα σημείο το οποίο με κάποιο τρόπο μπορεί να δώσει όλα τα υπόλοιπα σημεία της καμπύλης. Ο τρόπος αυτός είναι ο πολλαπλασιασμός του G με κάποιο ακέραιο αριθμό k. Παρακάτω θα δούμε αναλυτικά τι σημαίνει πολλαπλασιασμός σημείου με ακέραιο και τρόπους με τους οποίους γίνεται αποτελεσματικά.

6.3 Πρόσθεση σε Ελλειπτική Καμπύλη: Μια Γεωμετρική Προσέγγιση.

Αν P και Q είναι δύο σημεία της ελλειπτικής καμπύλης, τότε από την πρόσθεση τους προκύπτει ένα τρίτο σημείο, το R.

P + Q = R είναι η προσθετική ιδιότητα που ορίζεται γεωμετρικά.

Οι ομάδες ελλειπτικών καμπυλών είναι προσθετικές ομάδες. Δηλαδή, η βασική τους συνάρτηση είναι η πρόσθεση. Η πρόσθεση δύο σημείων σε μια ελλειπτική καμπύλη ορίζεται γεωμετρικά.

Το αρνητικό ενός σημείου $P = (x_P, y_P)$ είναι το συμμετρικό του ως προς τον άξονα των x: το σημείο –P είναι το $(x_P, -y_P)$. Παρατηρείστε επίσης (βλ. σχήμα 2), ότι για κάθε σημείο P σε μια ελλειπτική καμπύλη, το σημείο –P βρίσκεται επίσης πάνω στην ελλειπτική καμπύλη.

Η επιθυμητή ομάδα ορίζεται αν θεωρήσουμε ως άθροισμα των P και Q το συμμετρικό του R ως προς τον οριζόντιο άξονα, -R.

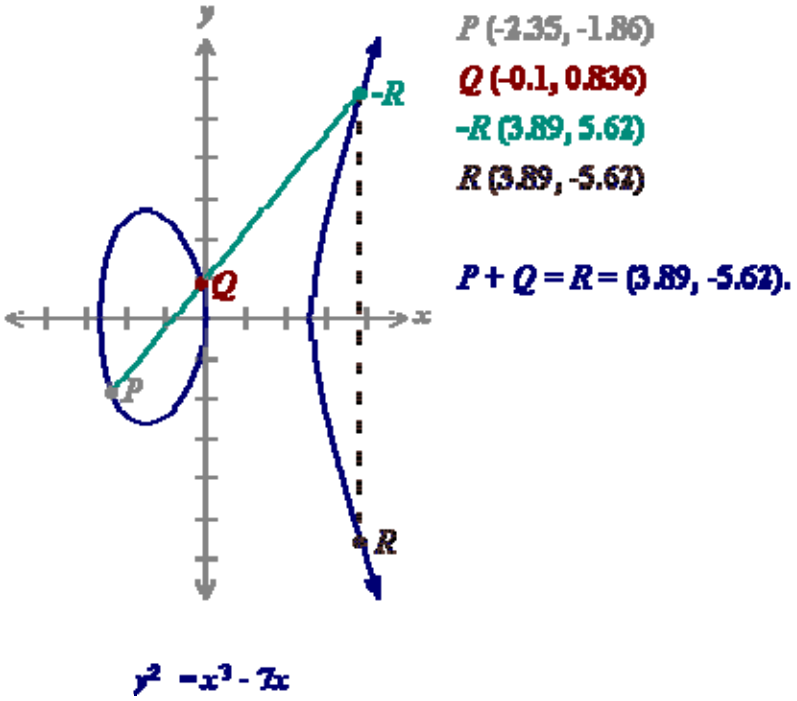
Αποδεικνύεται ότι αν η πρόσθεση οριστεί κατά αυτό τον τρόπο τότε [14]:

- Το άθροισμα δύο σημείων είναι σημείο της ελλειπτικής καμπύλης.
- Η πρόσθεση είναι προσεταιριστική πράξη.
- Το σημείο στο άπειρο είναι το ουδέτερο στοιχείο.
- Ορίζεται ο αντίστροφος ενός σημείου (το άλλο σημείο της καμπύλης που έχει την ίδια x - συντεταγμένη).
- Η πρόσθεση είναι αντιμεταθετική πράξη.

6.3.1 Προσθέτοντας ξεχωριστά σημεία P και Q.

Υποθέστε ότι τα P και Q είναι δύο ξεχωριστά σημεία σε μια ελλειπτική καμπύλη, και το P είναι διάφορο του Q. Για να προστεθούν τα σημεία P και Q, μια γραμμή σχεδιάζεται μεταξύ των δυο σημείων. Η γραμμή αυτή τέμνει την ελλειπτική καμπύλη σε ακόμα ένα σημείο, το –R. Το –R έχει το συμμετρικό του στον άξονα

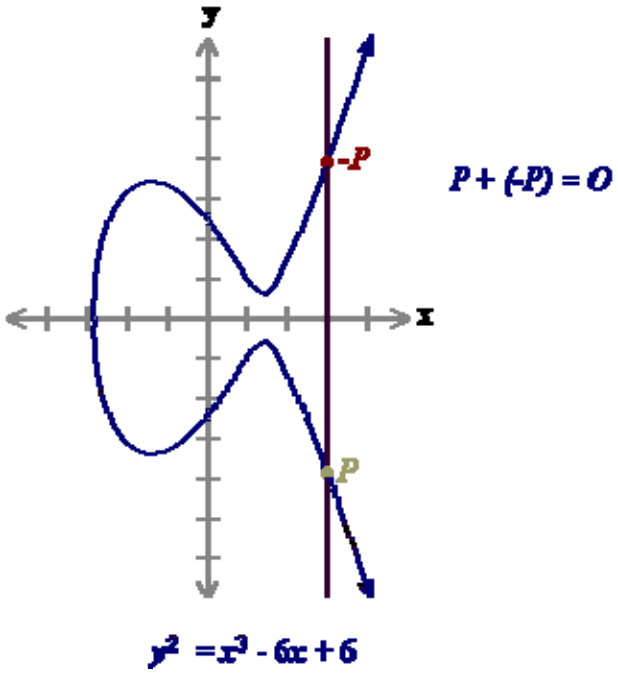
των x στο σημείο R. Ο κανόνας για την πρόσθεση σε μια ομάδα ελλειπτικής καμπύλης είναι P + Q = R. Για παράδειγμα:



Σχήμα 2: Προσθέτοντας ξεχωριστά σημεία P και Q.

6.3.2 Προσθέτοντας τα σημεία P και –P.

Η γραμμή μεταξύ των P και –P είναι μια κάθετη γραμμή η οποία δεν τέμνει την ελλειπτική καμπύλη σε κάποιο τρίτο σημείο. Έτσι, τα σημεία P και –P δε μπορούν να προστεθούν όπως προηγουμένως. Γι' αυτό το λόγο η ομάδα της ελλειπτικής καμπύλης συμπεριλαμβάνει το σημείο Ο. Εξ' ορισμού, P + (-P) = O στην ομάδα ελλειπτικής καμπύλης. Το Ο λέγεται ουδέτερο στοιχείο της πρόσθεσης (ή προσθετική ταυτότητα). Όλες οι ελλειπτικές καμπύλες έχουν ουδέτερο στοιχείο πρόσθεσης (βλ. σχήμα 3).

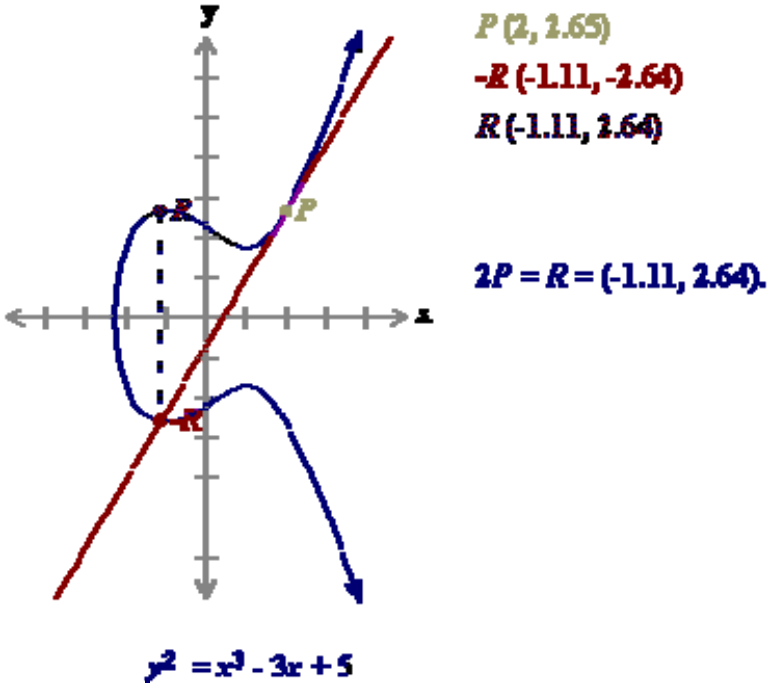


Σχήμα 3: Προσθέτοντας τα σημεία P και –P.

6.3.3 Διπλασιάζοντας το σημείο P.

Ο διπλασιασμός ενός σημείου P είναι ισοδύναμος με την πρόσθεση του σημείου στον ίδιο του τον εαυτό. Για να προστεθεί το σημείο P στον εαυτό του, μια εφαπτόμενη γραμμή στο σημείο P της καμπύλης σχεδιάζεται. Αν το y_P είναι διάφορο του 0, τότε η εφαπτόμενη γραμμή τέμνει την ελλειπτική καμπύλη ακριβώς σε ένα άλλο σημείο, το –R. Το –R έχει σαν συμμετρικό του ως προς τον άξονα των x το R. Η πράξη αυτή ονομάζεται διπλασιασμός του σημείου P (βλ. σχήμα 4). Ο κανόνας για τον διπλασιασμό ενός σημείου σε μια ομάδα ελλειπτικής καμπύλης ορίζεται ως εξής:

P + P = 2P = R



Σχήμα 4: Διπλασιάζοντας το σημείο P.

6.3.4 Διπλασιάζοντας το P αν $y_P = 0$.

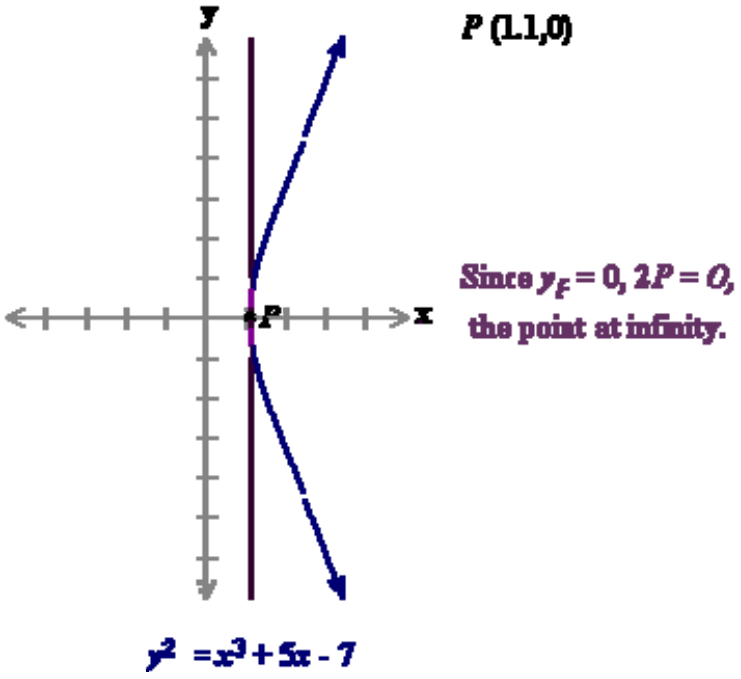
Η εφαπτόμενη στο P είναι πάντα κάθετη στον άξονα των x αν $y_P = 0$.

Αν ένα σημείο P είναι τέτοιο ώστε $y_P = 0$, τότε η εφαπτόμενη γραμμή της ελλειπτικής καμπύλης στο σημείο P είναι κάθετη και δεν τέμνει την ελλειπτική καμπύλη σε κανένα άλλο σημείο (βλ. σχήμα 5).

Εξ' ορισμού, 2P = O για ένα τέτοιο σημείο P.

Για να βρούμε το 3P σε αυτή τη περίπτωση, προσθέτουμε 2P + P. Αυτό μας κάνει P + O = P. Έτσι, 3P = P.

3P = P, 4P = O, 5P = P, 6P = O, 7P = P, κ.τ.λ.



Σχήμα 5: Διπλασιάζοντας το P αν $y_P = 0$.

6.4 Πρόσθεση σε Ελλειπτική Καμπύλη: Μια αλγεβρική Προσέγγιση.

Παρόλο που οι προηγούμενες γεωμετρικές περιγραφές των ελλειπτικών καμπυλών παρέχουν μια τέλεια μέθοδο αριθμητικής ερμηνείας της ελλειπτικής καμπύλης, δεν είναι αυτός ένας πρακτικός τρόπος για εφαρμογή αριθμητικών υπολογισμών. Αλγεβρικοί τύποι κατασκευάστηκαν για να υπολογίσουν αποτελεσματικά τη γεωμετρική αριθμητική.

6.4.1 Προσθέτοντας ξεχωριστά σημεία P και Q.

Όταν τα $P = (x_P, y_P)$ και $Q = (x_Q, y_Q)$ δεν είναι αρνητικά μεταξύ τους,

$P + Q = R$ όπου

$s = (y_P - y_Q) / (x_P - x_Q)$, $x_R = s^2 - x_P - x_Q$ και $y_R = -y_P + s(x_P - x_R)$

Σημειώστε ότι το s είναι η κλίση της γραμμής μεταξύ του P και Q.

6.4.2 Διπλασιάζοντας το σημείο P.

Όταν το y_P δεν είναι 0.

2P = R όπου

$s = (3x_P^2 + a) / (2y_P)$

$x_R = s^2 - 2x_P$ και $y_R = -y_P + s(x_P - x_R)$

Θυμίζουμε ότι το a είναι μία από τις παραμέτρους της ελλειπτικής καμπύλης και το s είναι η εφαπτόμενη στο σημείο P.

6.5 Παραδείγματα για την κατανόηση των ομάδων ελλειπτικών καμπυλών.

6.5.1 Ένα μοντέλο ελλειπτικής καμπύλης πραγματικών αριθμών.

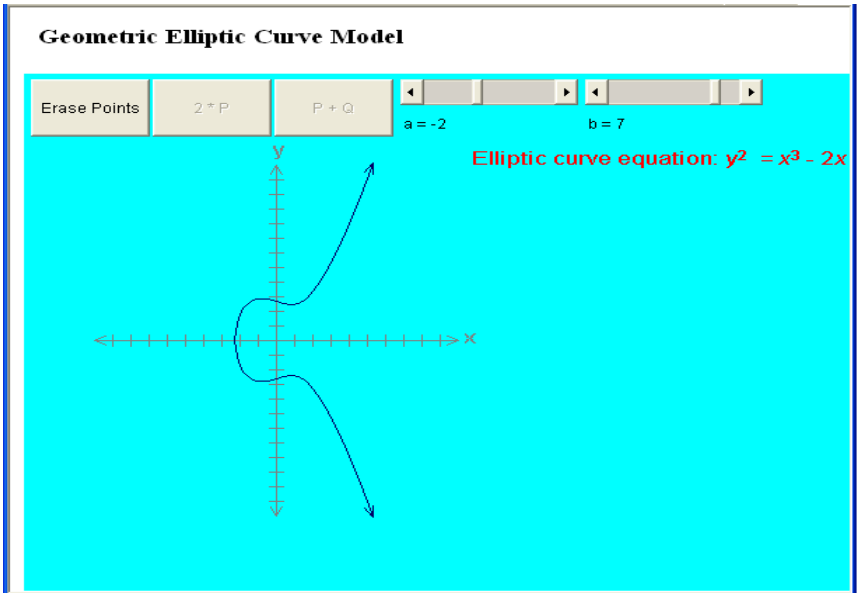
Το επόμενο μοντέλο χρησιμοποιείται για πειραματισμό με τη πρόσθεση σημείων διαφόρων ομάδων ελλειπτικών καμπυλών.

Για παραπάνω εξάσκηση μπορείτε να εξασκηθείτε με το javascript applet που βρίσκεται στην ιστοσελίδα:

http://www.certicom.com/ecc_tutorial/ecc_javaCurve.html

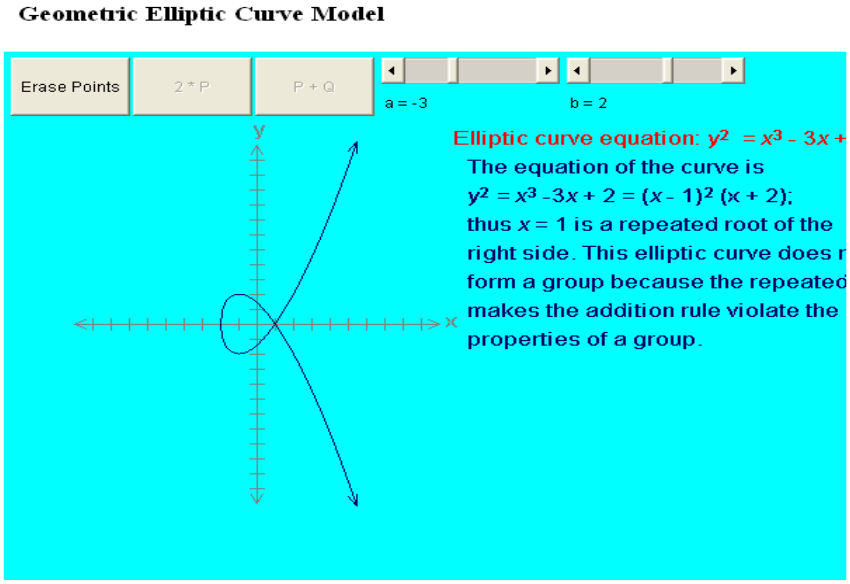
Τα σχήματα που ακολουθούν είναι από τη συγκεκριμένη ιστοσελίδα.

Αρχικά, δοκιμάζουμε για a = -2 και b = 7 και παίρνουμε την ελλειπτική καμπύλη που ακολουθεί.



Σχήμα 6: Ένα μοντέλο ελλειπτικής καμπύλης πραγματικών αριθμών που σχηματίζει ομάδα.

Αν δοκιμάσουμε για a = -3 και b = 2, θα δούμε ότι επειδή για x = 1 υπάρχει διπλή ρίζα στην ελλειπτική καμπύλη, αυτή δεν αποτελεί ομάδα διότι παραβιάζονται οι ιδιότητες της ομάδας εξαιτίας αυτού του γεγονότος (βλ. σχήμα 7).



Σχήμα 7: Ένα μοντέλο ελλειπτικής καμπύλης πραγματικών αριθμών που δε σχηματίζει ομάδα.

6.5.2 Εξάσκηση πάνω στις ομάδες ελλειπτικών καμπυλών.

Ομάδες Ελλειπτικών Καμπυλών πραγματικών αριθμών.

1.

Η εξίσωση ελλειπτικής καμπύλης πραγματικών αριθμών $y^2 = x^3 - 7x - 6$ καθορίζει μια ομάδα;

Ναι, αφού $4a^3 + 27b^2 = 4(-7)^3 + 27(-6)^2 = -400$

Η ισότητα $y^2 = x^3 - 7x – 6$ ορίζει μια ομάδα ελλειπτικής καμπύλης διότι το $4a^3 + 27b^2$ είναι διάφορο του μηδέν.

2.

Ποιο είναι το ουδέτερο στοιχείο της πρόσθεσης για τους κανονικούς ακέραιους;

Το ουδέτερο στοιχείο της πρόσθεσης για τους κανονικούς ακέραιους είναι το 0, αφού $x + 0 = x$ για κάθε ακέραιο.

3.

Ανήκει το σημείο (4, 7) στην ελλειπτική καμπύλη πραγματικών αριθμών $y^2 = x^3 - 5x + 5$;

Ναι, αφού η εξίσωση επαληθεύεται για x=4 και y=7:

$(7)^2 = (4)^3 - 5(4) + 5$

$49 = 64 - 20 + 5$

$49 = 49$

4.

Ποιοι είναι οι αρνητικοί αριθμοί των επόμενων σημείων ελλειπτικής καμπύλης πραγματικών αριθμών; P(-4, -6), Q(17, 0), R(3, 9), S(0, -4).

Ο αρνητικός αριθμός είναι το συμμετρικό σημείο στον άξονα των x. Έτσι

-P(-4, 6), -Q(17, 0), -R(3, -9), -S(0, 4)

5.

Στην ομάδα ελλειπτικής καμπύλης πραγματικών αριθμών που ορίζεται από την $y^2 = x^3 - 17x + 16$ ποιο είναι το P + Q αν P = (0, -4) και Q = (1, 0);

Από τον τύπο της πρόσθεσης:

$s = (y_P - y_Q) / (x_P - x_Q) = (-4 - 0) / (0 - 1) = 4$

$x_R = s^2 - x_P - x_Q = 16 - 0 - 1 = 15$

και

$y_R = -y_P + s(x_P - x_R) = 4 + 4(0 - 15) = -56$

Έτσι P + Q = (15, -56)

6.

Στην ομάδα ελλειπτικής καμπύλης πραγματικών αριθμών $y^2 = x^3 - 17x + 16$, ποιο είναι το 2P αν P = (4, 3,464);

Από τον τύπο του διπλασιασμού:

$s = (3x_P^2 + a) / (2y_P) = (3(4)^2 + (-17)) / 2(3,464) = 31 / 6,928 = 4,475$

$x_R = s^2 - 2x_P = (4,475)^2 - 2(4) = 20,022 - 8 = 12,022$

και

$y_R = -y_P + s(x_P - x_R) = -3,464 + 4,475(4 – 12,022) = - 3,464 – 35,898 = -39,362$

Έτσι 2P = (12,022, -39,362)

6.6 Ομάδες ελλειπτικής καμπύλης F_p.

Μια σημαντική ιδιότητα των ομάδων ελλειπτικών καμπυλών είναι ότι αυτές έχουν πεπερασμένο αριθμό σημείων.

Οι υπολογισμοί με πραγματικούς αριθμούς είναι αργοί και ανακριβής εξαιτίας του λάθους λόγω στρογγυλοποίησης. Οι κρυπτογραφικές εφαρμογές απαιτούν γρήγορη και ακριβής αριθμητική. Έτσι, οι ομάδες ελλειπτικών καμπυλών των πεπερασμένων σωμάτων F_p και F_{2^m} χρησιμοποιούνται συχνότερα.

Υπενθυμίζουμε ότι το σώμα F_p χρησιμοποιεί αριθμούς από το 0 έως το p – 1, και οι υπολογισμοί έχουν αποτέλεσμα το υπόλοιπο της διαίρεσης με το p. Για

παράδειγμα, στο F_{23} το σώμα αποτελείται από ακέραιους από το 0 μέχρι το 22, και κάθε πράξη μέσα στο σώμα έχει αποτέλεσμα έναν ακέραιο επίσης μεταξύ 0 και 22.

Μια ελλειπτική καμπύλη που υπόκεινται στο σώμα F_p μπορεί να σχηματιστεί διαλέγοντας τις μεταβλητές a και b από το σώμα F_p. Η ελλειπτική καμπύλη περιλαμβάνει όλα τα σημεία (x, y) που ικανοποιούν την εξίσωση modulo p (όπου x και y είναι αριθμοί στο F_p).

Για παράδειγμα: Η εξίσωση $y^2 \bmod p = x^3 + ax + b \bmod p$ παράγει το σώμα F_p αν a και b ανήκουν στο F_p.

Αν η $x^3 + ax + b$ δεν περιλαμβάνει επαναλαμβανόμενους παράγοντες (ή ισοδύναμα, αν το $4a^3 + 27b^2$ είναι διάφορο του 0), τότε η ελλειπτική καμπύλη μπορεί να σχηματίσει μια ομάδα. Μια ομάδα ελλειπτικής καμπύλης του F_p αποτελείται από σημεία της αντίστοιχης ελλειπτικής καμπύλης, μαζί με ένα ειδικό σημείο Ο που ονομάζεται σημείο στο άπειρο. Υπάρχουν πολλά πεπερασμένα σημεία σε μια τέτοια ελλειπτική καμπύλη.

6.6.1 Παράδειγμα Ομάδας Ελλειπτικής Καμπύλης στο F_p.

Παρατηρείστε τη φαινομενικά τυχαία διανομή των σημείων της ελλειπτικής καμπύλης στο F_p.

Σαν ένα μικρό παράδειγμα, σκεφτείτε μια ελλειπτική καμπύλη στο F_{23}. Με a = 1 και b = 0, η εξίσωση της ελλειπτικής καμπύλης είναι $y^2 = x^3 + x$. Το σημείο (9, 5) ικανοποιεί την εξίσωση αφού:

$y^2 \bmod p = x^3 + x \bmod p$

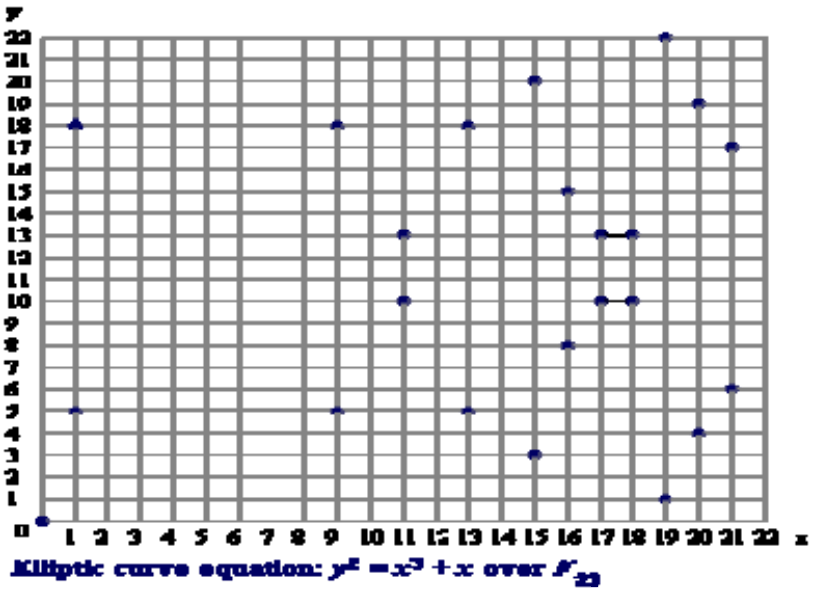
$25 \bmod 23 = 729 + 9 \bmod 23$

$25 \bmod 23 = 738 \bmod 23$

$2 = 2$

Τα 23 σημεία τα οποία ικανοποιούν την εξίσωση είναι τα 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)

Τα σημεία φαίνονται στο παρακάτω γράφημα (σχήμα 8):



Σχήμα 8: Τα σημεία μιας ομάδας ελλειπτικής καμπύλης στο F_p.

Σημειώστε ότι υπάρχουν δύο σημεία για κάθε τιμή x. Παρόλο που το γράφημα φαίνεται τυχαίο, υπάρχει συμμετρία στο y = 11,5. Θυμηθείτε ότι στις ελλειπτικές καμπύλες πραγματικών αριθμών υπάρχει ένα αρνητικό σημείο για κάθε σημείο, το οποίο είναι το συμμετρικό του στον άξονα των x. Στο F_{23}, η αρνητική τιμή της συνιστώσας του y modulo 23 μας δίνει τον θετικό αριθμό σαν τη διαφορά του από το 23.

$-P = (x_p, (-y_p \bmod 23))$

6.7 Αριθμητική σε μια Ομάδα Ελλειπτικής Καμπύλης F_p.

Αυτοί οι κανόνες είναι ίδιοι με αυτούς των ομάδων ελλειπτικών καμπυλών με πραγματικούς αριθμούς, με εξαίρεση ότι οι υπολογισμοί εκτελούνται σε modulo p.

Υπάρχουν αρκετές σημαντικές διαφορές ανάμεσα στις ομάδες ελλειπτικών καμπυλών F_p και σ' αυτές των πραγματικών αριθμών. Οι ομάδες ελλειπτικής καμπύλης στο F_p έχουν πεπερασμένο πλήθος σημείων, τα οποία αποτελούν απαραίτητη προϋπόθεση για κρυπτογραφική χρήση. Εφόσον αυτές οι καμπύλες αποτελούνται από αρκετά ξεχωριστά σημεία, δεν είναι ξεκάθαρο πώς θα συνδεθούν αυτά τα σημεία για να φτιάξουν μια γραφική παράσταση που να μοιάζει με καμπύλη.

Δεν είναι ξεκάθαρο πώς να εφαρμοστούν οι γεωμετρικές σχέσεις. Σαν αποτέλεσμα, η γεωμετρία που χρησιμοποιείται στις ομάδες ελλειπτικών καμπυλών πραγματικών αριθμών δε μπορεί να χρησιμοποιηθεί για τις ομάδες καμπυλών του σώματος F_p. Ωστόσο, οι κανόνες άλγεβρας για την αριθμητική μπορούν να υιοθετηθούν για τις ελλειπτικές καμπύλες στο F_p. Αντίθετα με τις ελλειπτικές καμπύλες με πραγματικούς αριθμούς, οι υπολογισμοί στο F_p δεν εμπλέκουν το λάθος της στρογγυλοποίησης, μια σημαντική ιδιότητα που απαιτείται από ένα κρυπτοσύστημα.

6.7.1 Προσθέτοντας μοναδικά σημεία P και Q.

Το αρνητικό σημείο του $P(x_p, y_p)$ είναι το σημείο $-P(x_p, -y_p \bmod p)$. Αν το P και το Q είναι μοναδικά σημεία τέτοια ώστε το P να είναι διάφορο του –Q, τότε:

$P + Q = R$ όπου
$s = (y_P - y_Q) / (x_P - x_Q) \bmod p$
$x_R = s^2 - x_P - x_Q \bmod p$ και $y_R = -y_P + s(x_P - x_R) \bmod p$
Σημειώστε ότι το s είναι η κλίση της γραμμής μεταξύ του P και του Q.

6.7.2 Διπλασιάζοντας το σημείο P.

Υποθέτοντας ότι το y_p δεν είναι 0,
$2P = R$ όπου
$s = (3x_P^2 + a) / (2y_P) \bmod p$
$x_R = s^2 - 2x_P \bmod p$ και $y_R = -y_P + s(x_P - x_R) \bmod p$

Θυμηθείτε ότι το a είναι ένας από τους παραμέτρους της εξίσωσης της ελλειπτικής καμπύλης και ότι το s είναι η κλίση της γραμμής μεταξύ του P και του Q.

6.8 Παραδείγματα για την κατανόηση των ομάδων ελλειπτικών καμπυλών στο F_p.

6.8.1 Ένα μοντέλο ελλειπτικής καμπύλης στο F_p.

Το επόμενο μοντέλο μπορεί να χρησιμοποιηθεί για την πρόσθεση σε διάφορες ομάδες ελλειπτικών καμπυλών.

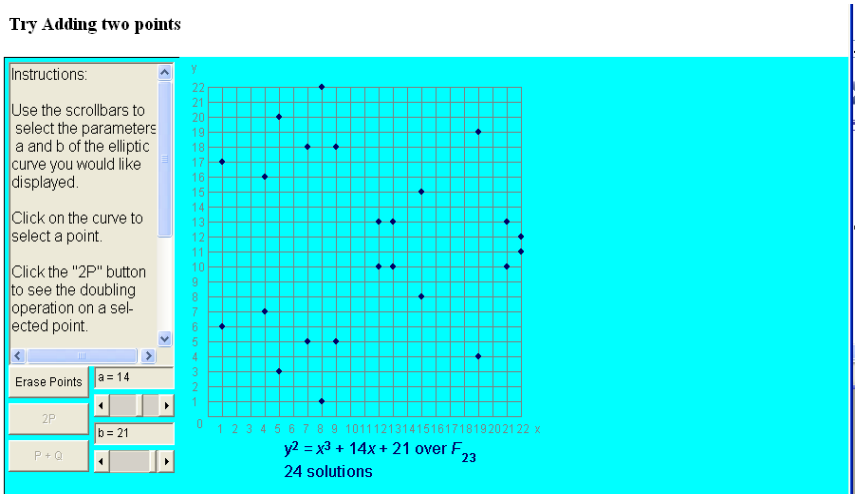
Πεπερασμένο Γεωμετρικό Μοντέλο Ελλειπτικής Καμπύλης: Για παραπάνω εξάσκηση μπορείτε να εξασκηθείτε με το javascript applet που βρίσκεται στο εξής site:

http://www.certicom.com/ecc_tutorial/ecc_twopoints.html

Δοκιμάσαμε τα επόμενα:

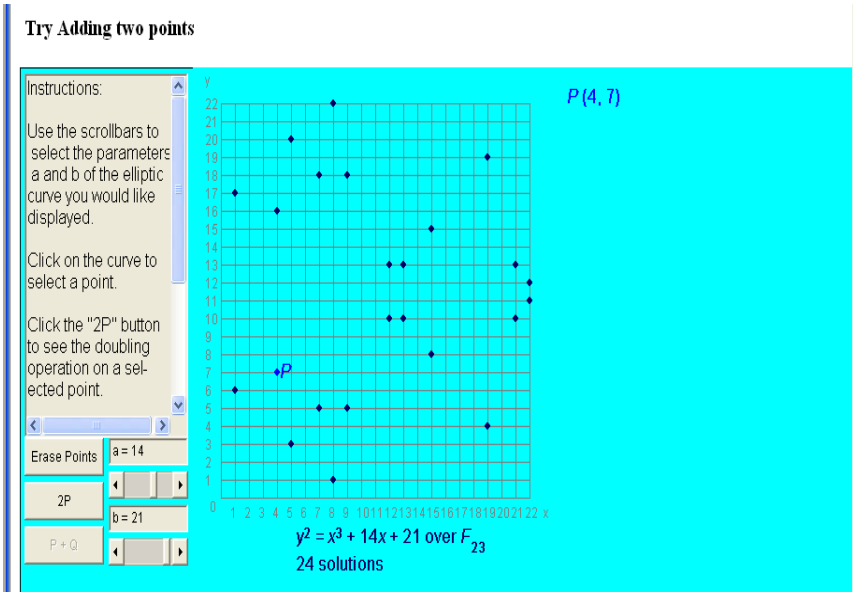
1. Αλλάξαμε τις μεταβλητές a και b για να δούμε τα σημεία που προκύπτουν στην καμπύλη.
2. Διαλέξαμε ένα σημείο P στη καμπύλη, και μετά διαλέξαμε άλλο ένα σημείο Q. Τα προσθέσαμε μαζί.
3. Διαλέξαμε ένα σημείο P και μετά το διπλασιάσαμε.
4. Δοκιμάσαμε να διπλασιάσουμε το P όταν $y_p = 0$ (2P = 0, αφού $y_p = 0$).
5. Δοκιμάσαμε να προσθέσουμε δύο σημεία με την ίδια τιμή x (P + Q = 0, αφού P = -Q).

Έστω ότι έχουμε την ελλειπτική καμπύλη $y^2 = x^3 + 14x + 21$ στο F_{23}.



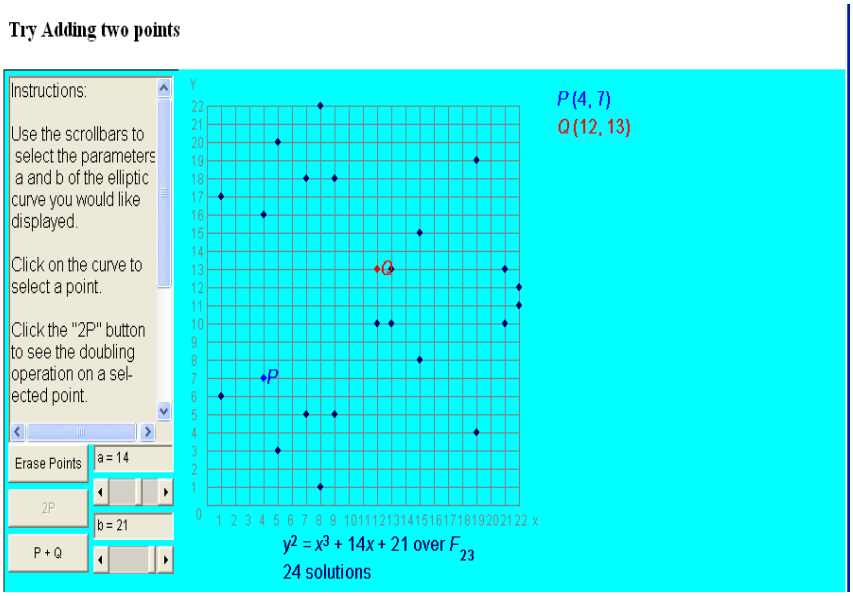
Σχήμα 9: Η ελλειπτική καμπύλη $y^2 = x^3 + 14x + 21$ στο F_{23}.

Έστω ότι επιλέγουμε το σημείο P(4, 7) που ανήκει σε αυτή. Ακολουθεί το σχήμα 10.



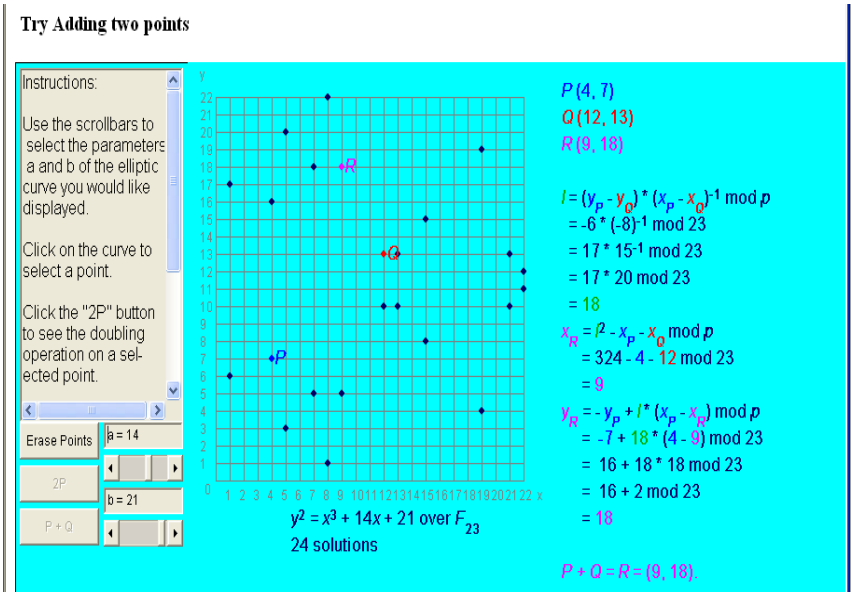
Σχήμα 10: Το σημείο P(4, 7) στην ελλειπτική καμπύλη $y^2 = x^3 + 14x + 21$ στο F_{23}.

Έστω ότι επιλέγουμε και το σημείο Q(12, 13). Ακολουθεί το σχήμα 11.



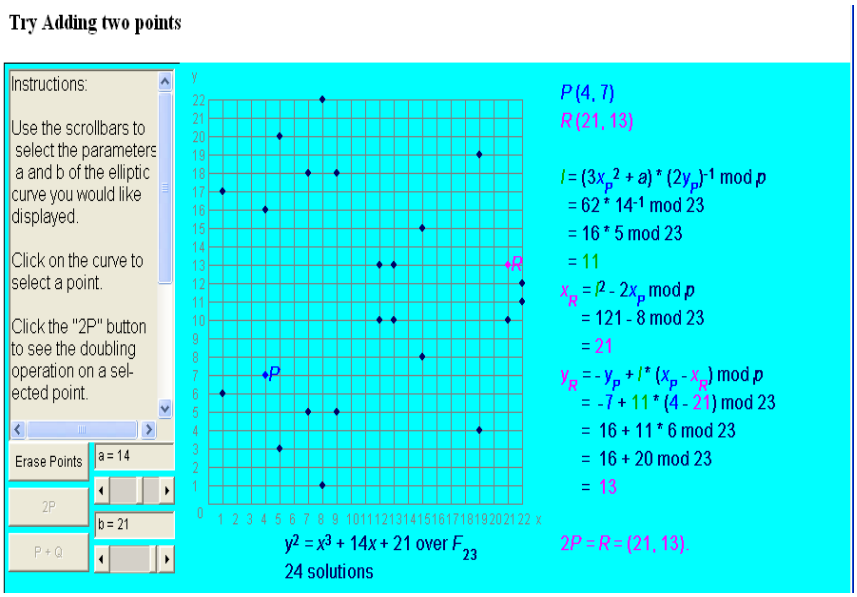
Σχήμα 11: Τα σημεία P(4, 7) και Q(12, 13) στην ελλειπτική καμπύλη $y^2 = x^3 + 14x + 21$ στο F_{23}.

Μετά τη πρόσθεση των 2 σημείων θα έχουμε (σχήμα 12):



Σχήμα 12: Η πρόσθεση των 2 σημείων P(4, 7) και Q(12, 13) στην ελλειπτική καμπύλη $y^2 = x^3 + 14x + 21$ στο F_{23}.

Επίσης, σε περίπτωση που θέλουμε να βρούμε το 2P θα έχουμε (σχήμα 13):



Σχήμα 13: Εύρεση του 2P στην ελλειπτική καμπύλη $y^2 = x^3 + 14x + 21$ στο F_{23}.

6.8.2 Εξάσκηση πάνω στις ομάδες ελλειπτικών καμπυλών στο F_p.

1. Η εξίσωση ελλειπτικής καμπύλης $y^2 = x^3 + 10x + 5$ ορίζει μια ομάδα στο F_{17};

Όχι, αφού:

$4a^3 + 27b^2 \bmod p$

$= 4(10)^3 + 27(5)^2 \bmod 17$

$= 4675 \bmod 17$

$= 0$

Έτσι η ελλειπτική καμπύλη δεν ορίζει ομάδα διότι το $4a^3 + 27b^2 \bmod p$ είναι 0.

2. Τα σημεία P(2, 0) και Q(6, 3) βρίσκονται στην ελλειπτική καμπύλη $y^2 = x^3 + x + 7$ του σώματος F_{17};

Το σημείο P(2, 0) βρίσκεται στην ελλειπτική καμπύλη αφού ισχύει η ισότητα:

$(0)^2 \bmod 17 = (2)^3 + 2 + 7 \bmod 17$

$0 \bmod 17 = 17 \bmod 17$

$0 = 0.$

Αντίθετα το σημείο Q(6, 3), δεν βρίσκεται στην ελλειπτική καμπύλη εφόσον η ισότητα δεν επαληθεύεται:

$(3)^2 \bmod 17 = (6)^3 + 6 + 7 \bmod 17$

$9 \bmod 17 = 229 \bmod 17$

$9 = 8$, δεν ισχύει.

3. Ποια είναι τα αρνητικά σημεία των παρακάτω σημείων της ίδιας ελλειπτικής καμπύλης του F_{17}; P(5, 8), Q(3, 0), R(0, 6)

Το αρνητικό σημείο ενός σημείου $P = (x_P, y_P)$ είναι το σημείο $-P = (x_P, -y_P \bmod p)$. Έτσι, αυτά θα είναι αντίστοιχα: -P(5, 9), -Q(3, 0), -R(0, 11)

4. Στην ομάδα ελλειπτικής καμπύλης που ορίζεται από την εξίσωση $y^2 = x^3 + x + 7$ στο F_{17}, πόσο κάνει το P + Q αν το P = (2, 0) και το Q = (1, 3);

$s = (y_P - y_Q) / (x_P - x_Q) \bmod p = (-3) / 1 \bmod 17 = -3 \bmod 17 = 14$

$x_R = s^2 - x_P - x_Q \bmod p = 196 - 2 - 1 \bmod 17 = 193 \bmod 17 = 6$

$y_R = -y_P + s(x_P - x_R) \bmod p = 0 + 14(2 - 6) \bmod 17 = -56 \bmod 17 = 12$

Έτσι P + Q = (6, 12).

5. Στην ομάδα ελλειπτικής καμπύλης που ορίζεται από τη $y^2 = x^3 + x + 7$ στο F_{17}, πόσο κάνει το 2P αν το P = (1, 3);

$s = (3x_P^2 + a) / (2y_P) \bmod p = (3+1) \cdot 6^{-1} \bmod 17 = 4 \cdot 3 \bmod 17 = 12$

$x_R = s^2 - 2x_P \bmod p = 144 - 2 \bmod 17 = 142 \bmod 17 = 6$

$y_R = -y_P + s(x_P - x_R) \bmod p = -3 + 12 \cdot (1 - 6) \bmod 17 = -63 \bmod 17 = 5$

Έτσι 2P = (6, 5)

6.9 Ομάδες Ελλειπτικών Καμπυλών στο F_2^m.

Υπάρχουν πολλά πεπερασμένα σημεία σε μια καμπύλη του σώματος F_{2^m}. Οι παράγοντες του σώματος F_{2^m} αποτελούν μια αλυσίδα από m bits. Οι κανόνες αριθμητικής στο F_{2^m} μπορούν να καθοριστούν από πολυωνυμική αναπαράσταση. Εφόσον, το F_{2^m} λειτουργεί σαν αλυσίδα bit, οι υπολογιστές μπορούν να εκτελέσουν την αριθμητική σε αυτό το σώμα αρκετά αποδοτικά.

Μια ελλειπτική καμπύλη στο F_{2^m} σχηματίζεται διαλέγοντας τους παράγοντες a και b από το F_{2^m} (με μοναδικό όρο ότι το b είναι διάφορο του 0). Έχοντας στο σώμα F_{2^m} το χαρακτηριστικό ψηφίο 2 πάντα, η εξίσωση της ελλειπτικής καμπύλης είναι κατάλληλη για δυαδική αναπαράσταση:

$$y^2 + xy = x^3 + ax^2 + b$$

Η ελλειπτική καμπύλη περιλαμβάνει όλα τα σημεία (x, y) τα οποία ικανοποιούν την εξίσωση της ελλειπτικής καμπύλης στο F_{2^m} (όπου τα x και y είναι στοιχεία του F_{2^m}). Μια ομάδα ελλειπτικής καμπύλης στο F_{2^m} αποτελείται από τα σημεία της αντίστοιχης ελλειπτικής καμπύλης, μαζί με το σημείο στο άπειρο, το O. Υπάρχουν πολλά πεπερασμένα σημεία σε κάθε τέτοια καμπύλη.

6.9.1 Παράδειγμα ομάδας ελλειπτικής καμπύλης στο F_2^m.

Η πρόσθεση με αλυσίδες bit ελέγχεται από μια λειτουργία XOR. Σαν ένα μικρό παράδειγμα, θεωρήστε το σώμα F_{2^4}, το οποίο ορίζεται με την πολυωνυμική αναπαράσταση με το μη αναγόμενο πολυώνυμο $f(x) = x^4 + x + 1$.

Το στοιχείο g = (0010) είναι γεννήτορας της ομάδας. Οι δυνάμεις του g είναι:

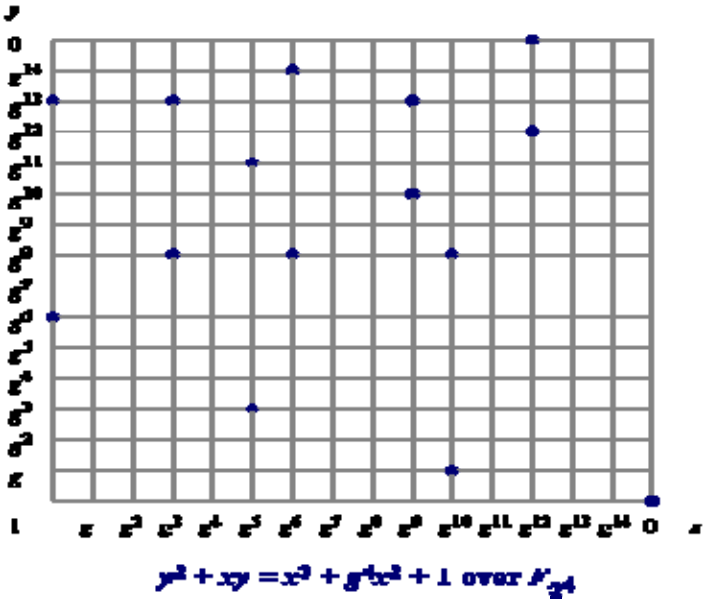
$g^0 = (0001)$ $g^1 = (0010)$ $g^2 = (0100)$ $g^3 = (1000)$ $g^4 = (0011)$ $g^5 = (0110)$
$g^6 = (1100)$ $g^7 = (1011)$ $g^8 = (0101)$ $g^9 = (1010)$ $g^{10} = (0111)$ $g^{11} = (1110)$
$g^{12} = (1111)$ $g^{13} = (1101)$ $g^{14} = (1001)$ $g^{15} = (0001)$

Ο πολλαπλασιασμός μεταξύ δύο αριθμών του σώματος αυτού γίνεται όπως είδαμε στην θεωρία των αριθμών, πάντα με modulo το πολυώνυμο $f(x) = x^4 + x + 1$. Σε μια αληθινή κρυπτογραφική εφαρμογή, η παράμετρος m πρέπει να είναι αρκετά μεγάλη ώστε να αποκλείσει την εύκολη δημιουργία ενός τέτοιου πίνακα αλλιώς το κρυπτοσύστημα θα σπάσει. Σύμφωνα με τη σημερινή εμπειρία, το m = 160 θα ήταν μια καλή επιλογή. Ο πίνακας επιτρέπει τη χρήση της σημειογραφίας του γεννήτορα (g^e) και όχι τόσο τη χρήση της σημειογραφίας της αλυσίδας bit, η οποία χρησιμοποιείται στο παρακάτω παράδειγμα. Επίσης, χρησιμοποιώντας σημειογραφία του γεννήτορα επιτρέπεται ο πολλαπλασιασμός χωρίς αναφορά στο μη αναγόμενο πολυώνυμο $f(x) = x^4 + x + 1$.

Έστω ότι έχουμε την ελλειπτική καμπύλη $y^2 + xy = x^3 + g^4x^2 + 1$. Εδώ είναι $a = g^4$ και $b = g^0 = 1$. Το σημείο (g^5, g^3) ικανοποιεί την εξίσωση στο F_{2^m}:

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

Τα 15 σημεία που επαληθεύουν την εξίσωση είναι:
$(1, g^{13})$ (g^3, g^{13}) (g^5, g^{11}) (g^6, g^{14}) (g^9, g^{13}) (g^{10}, g^8) (g^{12}, g^{12})
$(1, {}^{g6})$ $(g^{3,}\, g^8)$ (g^5, g^3) (g^6, g^8) (g^9, g^{10}) (g^{10}, g) $(g^{12}, 0)$ $(0, 1)$
Αυτά τα σημεία παρουσιάζονται παρακάτω στο σχήμα 14.



Σχήμα 14: Ελλειπτική καμπύλη $y^2 + xy = x^3 + g^4x^2 + 1$ στο F_{2^4} .

6.10 Αριθμητική σε μια Ομάδα Ελλειπτικής Καμπύλης F_2^m.

Οι ομάδες ελλειπτικής καμπύλης στο σώμα F_{2^m} έχουν πεπερασμένο αριθμό σημείων, και η αριθμητική τους δεν εμπλέκεται με το λάθος λόγω στρογγυλοποίησης. Το γεγονός αυτό συνδυασμένο με τη δυαδική φύση του σώματος, κάνει την αριθμητική του F_{2^m} να μπορεί να εκτελεστεί εύκολα από έναν υπολογιστή.

6.10.1 Προσθέτοντας τα σημεία P και Q.

Το αρνητικό του σημείου $P = (x_P, y_P)$ είναι το σημείο $-P = (x_P, x_P + y_P)$. Αν το P και το Q είναι ξεχωριστά σημεία τέτοια ώστε το P είναι διάφορο του –Q, τότε $P + Q = R$ όπου

$s = (y_P - y_Q) / (x_P + x_Q)$

$x_R = s^2 + s + x_P + x_Q + a$ και $y_R = s(x_P + x_R) + x_R + y_P$

Όπως με τις ομάδες ελλειπτικών καμπυλών πραγματικών αριθμών, P + (-P) = O, το σημείο στο άπειρο. Επίσης, P + O = P για όλα τα σημεία P σε μια ομάδα ελλειπτικής καμπύλης.

6.10.2 Διπλασιάζοντας το σημείο P.

Αν το $x_P = 0$, τότε 2P = O

Θεωρώντας ότι το x_P δεν είναι 0,

2P = R όπου

$s = x_P + y_P / x_P$

$x_R = s^2 + s + a$ και $y_R = x_P^2 + (s+1) \cdot x_R$

Θυμηθείτε ότι το a είναι ένας από τους παραμέτρους της ελλειπτικής καμπύλης και ότι το s είναι η κλίση της γραμμής που τέμνει τα P και Q.

6.11 Παραδείγματα για την κατανόηση των ομάδων ελλειπτικών καμπυλών στο F_2^m.

6.11.1 Ένα μοντέλο ελλειπτικής καμπύλης στο F_2^m.

Το επόμενο μοντέλο μπορεί να χρησιμοποιηθεί σαν πείραμα στην πρόσθεση και στο διπλασιασμό σε ορισμένες ομάδες ελλειπτικών καμπυλών του F_{2^4}.

Πεπερασμένο Γεωμετρικό Μοντέλο Ελλειπτικής Καμπύλης στο F_{2^4}. Το συγκεκριμένο applet βρίσκεται στο εξής site:

http://www.certicom.com/ecc_tutorial/ecc_javaCurve.html

Δοκιμάσαμε τα επόμενα πειράματα:

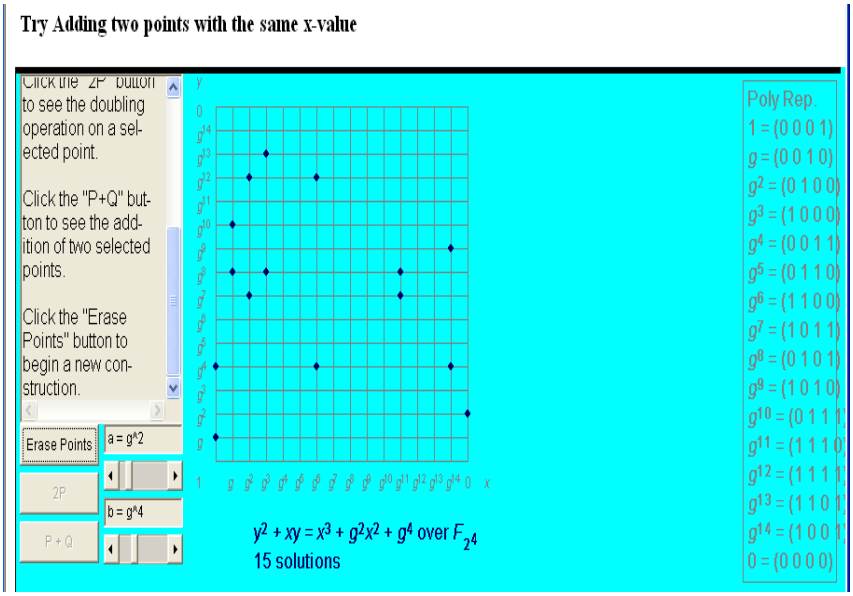
1. Αλλάξαμε τις μεταβλητές a και b για να βρούμε τα σημεία που προκύπτουν στη καμπύλη.
2. Επιλέξαμε ένα σημείο P στη καμπύλη, μετά επιλέξαμε ένα σημείο Q, και τέλος τα προσθέσαμε μαζί.
3. Επιλέξαμε ένα σημείο P στη καμπύλη και μετά το διπλασιάσαμε.

4. Δοκιμάσαμε να διπλασιάσουμε το P για g^0.

5. Δοκιμάσαμε να προσθέσουμε δύο σημεία που έχουν ίδια τιμή x.

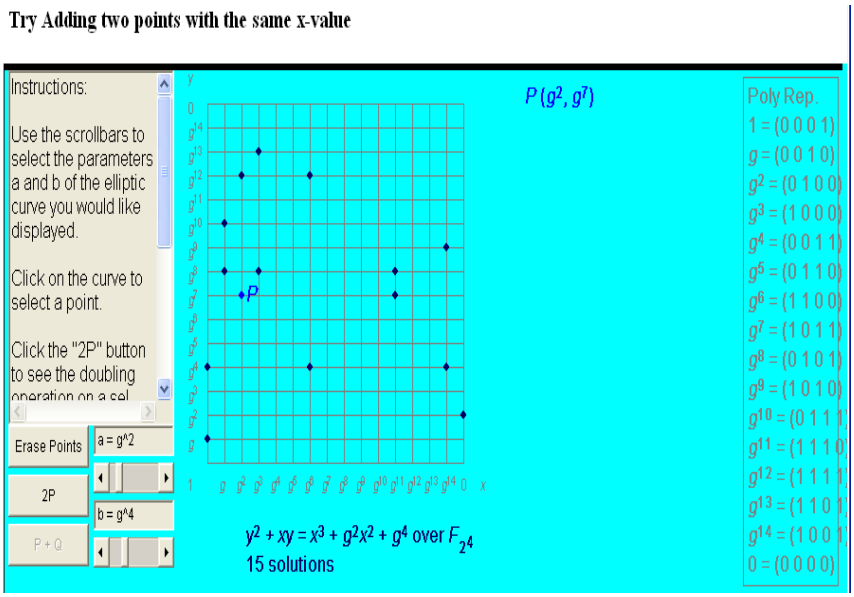
Στο σχήμα 15 έχουμε τα σημεία της ελλειπτικής καμπύλης:

$y^2 + xy = x^3 + g^2x^2 + g^4$ στο F_{2^4} .



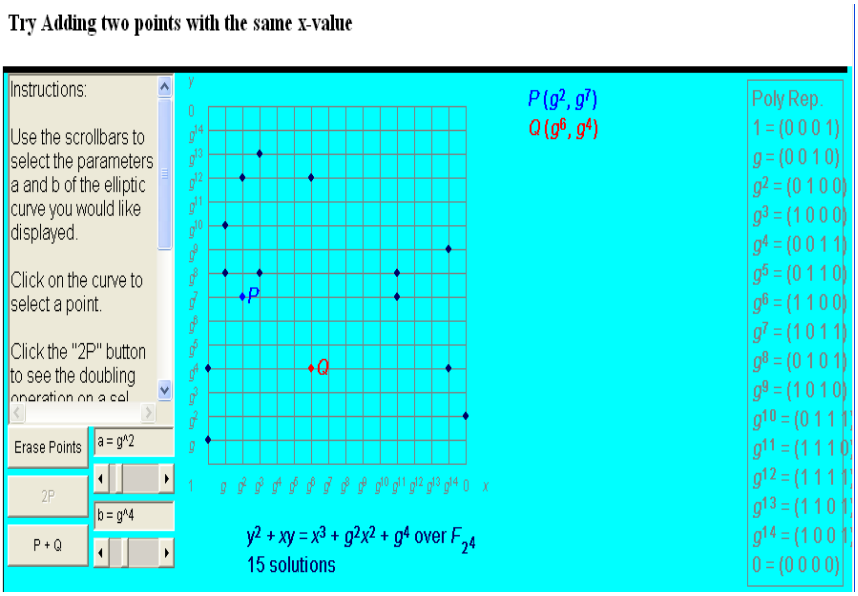
Σχήμα 15: Τα στοιχεία της ελλειπτικής καμπύλης $y^2 + xy = x^3 + g^4x^2 + 1$ στο F_{2^4} .

Έστω ότι επιλέγουμε το σημείο $P(g^2, g^7)$. Ακολουθεί το σχήμα 16.



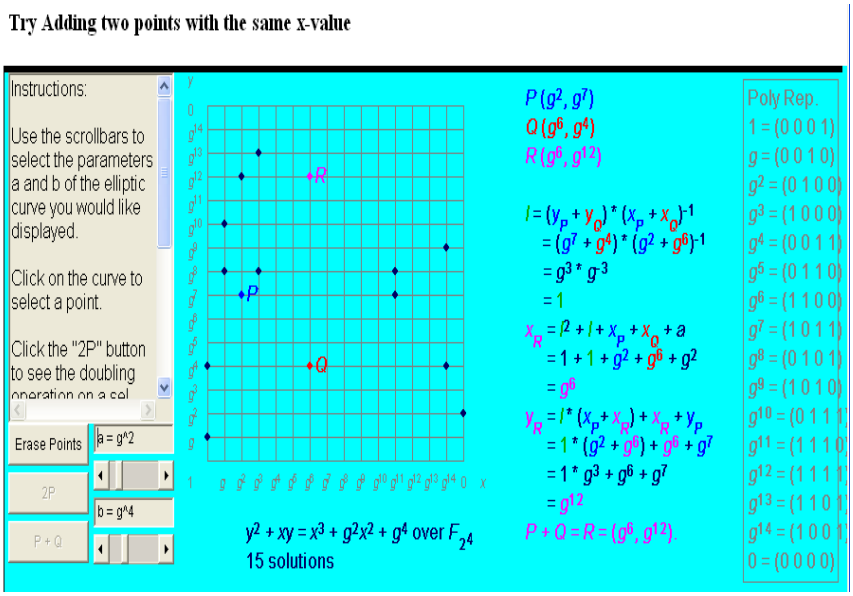
Σχήμα 16: Το σημείο $P(g^2, g^7)$ της ελλειπτικής καμπύλης $y^2 + xy = x^3 + g^4x^2 + 1$ στο F_{2^4} .

Έστω ότι επιλέγουμε το σημείο $Q(g^6, g^4)$. Ακολουθεί το σχήμα 17.



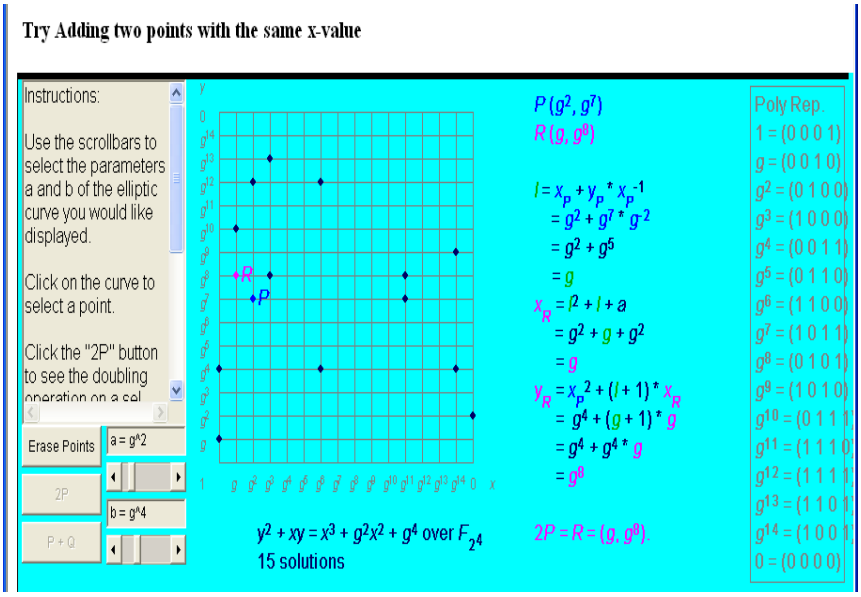
Σχήμα 17: Τα σημεία $P(g^2, g^7)$ και $Q(g^6, g^4)$ της ελλειπτικής καμπύλης $y^2 + xy = x^3 + g^4x^2 + 1$ στο F_{2^4} .

Μετά τη πρόσθεση των 2 σημείων θα έχουμε (σχήμα 18):



Σχήμα 18: Η πρόσθεση των σημείων $P(g^2, g^7)$ και $Q(g^6, g^4)$ της ελλειπτικής καμπύλης $y^2 + xy = x^3 + g^4x^2 + 1$ στο F_{2^4} .

Επίσης, σε περίπτωση που θέλουμε να βρούμε το 2P θα έχουμε (σχήμα 19):



Σχήμα 19: Εύρεση του 2P στην ελλειπτική καμπύλη $y^2 + xy = x^3 + g^4x^2 + 1$ στο F_{2^4} .

6.11.2 Εξάσκηση πάνω στις ομάδες ελλειπτικών καμπυλών στο F_2^m.

Ομάδες Ελλειπτικών Καμπυλών στο F_{2^m} .

1. Η εξίσωση ελλειπτικής καμπύλης $y^2 + xy = x^3 + g^5x^2 + g^6$ ορίζει μια ομάδα στο F_{2^3} ;

Αφού η παράμετρος b = 6 δεν είναι 0, η εξίσωση $y^2 + xy = x^3 + g^5x^2 + g^6$ ορίζει μια ομάδα στο F_{2^3} .

2. Τα σημεία $P(g^3, g^6)$, $Q(g^5, g^2)$ ανήκουν στην ελλειπτική καμπύλη $y^2 + xy = x^3 + g^2x^2 + g^6$ στο F_{2^3} ;

Το σημείο $P(g^3, g^6)$ ανήκει στην ελλειπτική καμπύλη $y^2 + xy = x^3 + g^2x^2 + g^6$ στο σώμα F_{2^3} , αφού επαληθεύει την εξίσωση:

$(g^6)^2 + (g^3)(g^6) = (g^3)^3 + g^2(g^3)^2 + g^6$ (Οι παράγοντες ανάγονται σε modulo g^7)

$g^5 + g^2 = g^2 + g + g^6$

$(111) + (100) = (100) + (010) + (101)$

$(011) = (011)$

$g^3 = g^3$

Όμως, το σημείο $Q(g^5, g^2)$ δεν ανήκει στην ελλειπτική καμπύλη, αφού δεν επαληθεύει την εξίσωση:

$(g^2)^2 + (g^5)(g^2) = (g^5)^3 + g^2(g^5)^2 + g^6$

$g^4 + 1 = g + g^5 + g^6$

$(110) + (001) = (001) + (111) + (101)$

$(111) = (000)$

$g^5 = 0$ δεν ισχύει.

3. Ποια είναι τα αρνητικά σημεία των παρακάτω σημείων της ελλειπτικής καμπύλης στο F_{2^3} $(P(g^3, g^6), Q(g, 0), R(0, g^3))$;

Τα αρνητικά σημεία ορίζονται ως $(x_P, x_P + y_P)$

$-P = (g^3, g^3 + g^6) = (g^3, g^4)$

$-Q = (g, g + 0) = (g, g)$

$-R = (0, 0 + g^3) = (0, g^3)$

4. Στην ομάδα ελλειπτικής καμπύλης $y^2 + xy = x^3 + g^2x^2 + g^6$ στο F_{2^3}, πόσο κάνει $P + Q$ αν $P = (g^2, g^6)$ και $Q = (g^5, g^5)$;

$P + Q = R$ όπου:

$s = (y^P - y^Q) / (x^P + x^Q) = (g^6 + g^5) / (g^2 + g^5) = g / g^3 = g\text{-}2 = g^5$

$x^R = s^2 + s + x^P + x^Q + a = g^3 + g^5 + g^2 + g^5 + g^2 = g^3$

$y^R = s(x^P + x^R) + x^R + y^P = g^5 (g^2 + g^3) + g^5 + g^6 = g^5\, g^5 + g^3 + g^6 = g^3 + g^3 + g^6 = g^6$

Έτσι $P + Q = (g^3, g^6)$

5. Στην ομάδα ελλειπτικής καμπύλης $y^2 + xy = x^3 + g^2x^2 + g^6$ στο F_{2^3}, πόσο κάνει $2P$ αν $P = (g^3, g^4)$;

$2P = R$ όπου:

$s = x^P + y^P / 2x^P = g^3 + g^4 / g^3 = g^3 + g = 1$

$x^R = s^2 + s + a = 1 + 1 + g^2 = g^2$

$y^R = x^P + (s + 1)\, x^R = g^3 + 0\, g^2 = g^3$

Έτσι $2P = (g^2, g^3)$

6.12 Εισαγωγή στο πρόβλημα του διακριτού λογαρίθμου.

Η δημιουργία κάποιου κρυπτοσυστήματος είναι ένα δύσκολο μαθηματικό πρόβλημα το οποίο λύνεται αρκετά δύσκολα. Το πρόβλημα του διακριτού λογαρίθμου είναι η βάση για την ασφάλεια πολλών κρυπτοσυστημάτων συμπεριλαμβανομένου του Κρυπτοσυστήματος των Ελλειπτικών Καμπυλών. Πιο συγκεκριμένα, η ECC (Elliptic Curve Cryptography) βασίζεται στη δυσκολία του Προβλήματος του Διακριτού Λογαρίθμου στις Ελλειπτικές Καμπύλες ECDLP (Elliptic Curve Discrete Logarithm Problem).

Θυμηθείτε ότι εξετάσαμε γεωμετρικά δύο καθορισμένες πράξεις σε συγκεκριμένες ομάδες ελλειπτικών καμπυλών. Αυτές οι δύο πράξεις ήταν η πρόσθεση σημείων και ο διπλασιασμός σημείων. Επιλέγοντας ένα σημείο P από μια ομάδα ελλειπτικής καμπύλης, μπορούμε να βρούμε το διπλάσιο του παίρνοντας το σημείο 2P. Μετά μπορούμε να προσθέσουμε ξανά το σημείο P στο σημείο 2P και να πάρουμε το σημείο 3P. Ο καθορισμός του σημείου nP με αυτό τον τρόπο αναφέρεται ως Βαθμωτός Πολλαπλασιασμός ενός σημείου. Το ECDLP βασίζεται στη δυσκολία εύρεσης των παραγόμενων στοιχείων του βαθμωτού πολλαπλασιασμού.

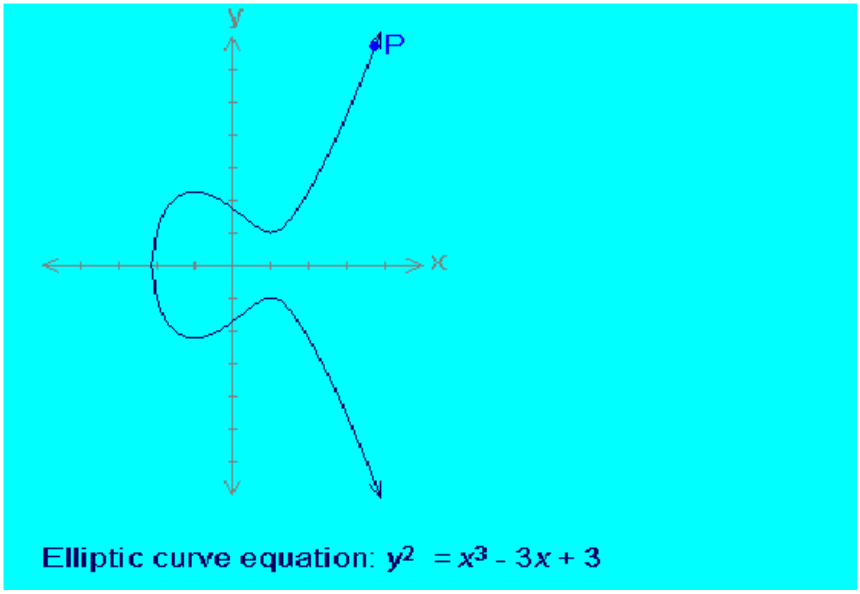
6.12.1 Βαθμωτός Πολλαπλασιασμός.

Στην ιστοσελίδα http://www.certicom.com/ecc_tutorial/ecc_tut_5_3_1.html υπάρχει κάποιο java applet το οποίο παρουσιάζει το βαθμωτό πολλαπλασιασμό μέσα από ένα συνδυασμό διπλασιασμού και προσθέσεων κάποιων σημείων.

Ενώ είναι σύνηθες να χρησιμοποιούμε το σύμβολο της πρόσθεσης για να περιγράψουμε μια ομάδα ελλειπτικής καμπύλης, η χρήση του συμβόλου του πολλαπλασιασμού είναι περισσότερο διαισθητική. Συγκεκριμένα, σκεφτείτε την πράξη του «βαθμωτού πολλαπλασιασμού» με το σύμβολο της πρόσθεσης: Έτσι, υπολογίζουμε το kP προσθέτοντας μαζί k φορές το σημείο P. Χρησιμοποιώντας το

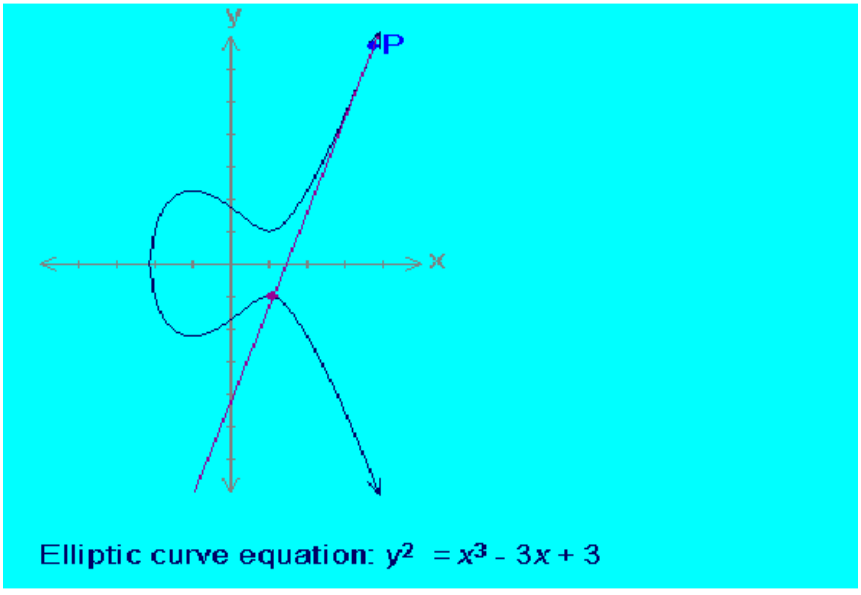
σύμβολο του πολλαπλασιασμού, η πράξη αυτή είναι σαν να πολλαπλασιάζουμε μαζί k αντίγραφα του σημείου P, παίρνοντας το σημείο $P \cdot P \cdot P \cdot P \cdot .... \cdot P = Pk$.

Παρακάτω ακολουθούν κάποια ενδεικτικά σχήματα της εφαρμογής για την εύρεση του διακριτού λογαρίθμου:



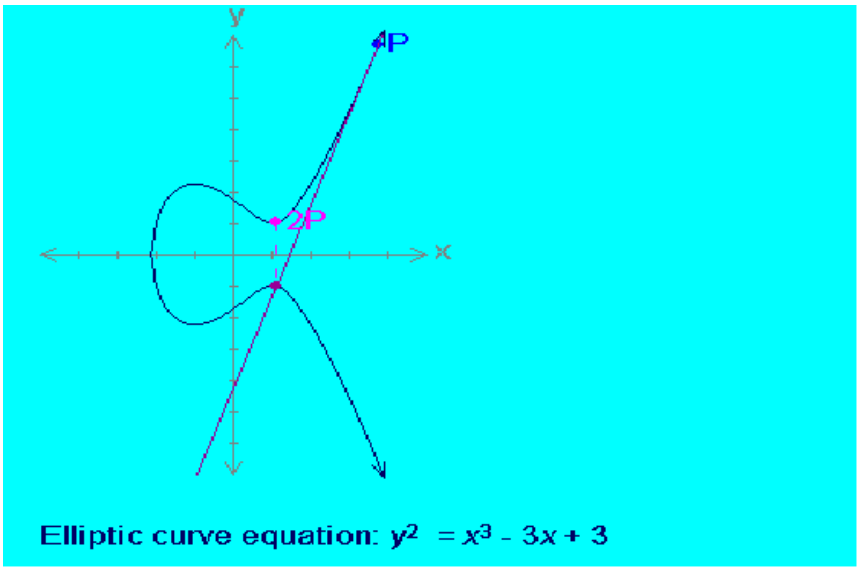
Σχήμα 20: Η ελλειπτική καμπύλη $y^2 = x^3 - 3x + 3$.

Παίρνουμε ένα σημείο P πάνω στην ελλειπτική καμπύλη και φέρουμε την εφαπτόμενη στο σημείο αυτό.



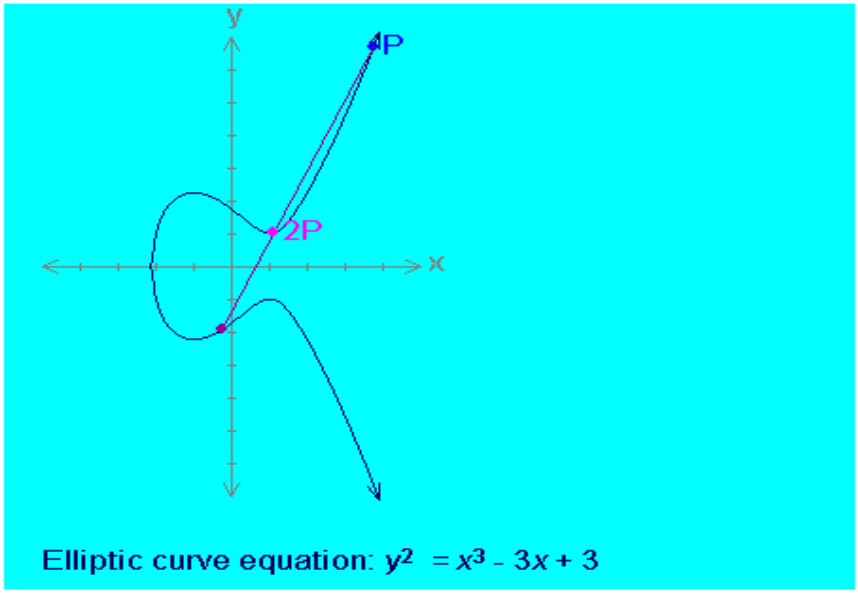
Σχήμα 21: Η εφαπτόμενη της ελλειπτική καμπύλη $y^2 = x^3 - 3x + 3$ στο σημείο P.

Στη συνέχεια, από το σημείο στο οποίο τέμνεται η εφαπτόμενη από το P με την ελλειπτική καμπύλη φέρνουμε την κάθετο στον άξονα των x.



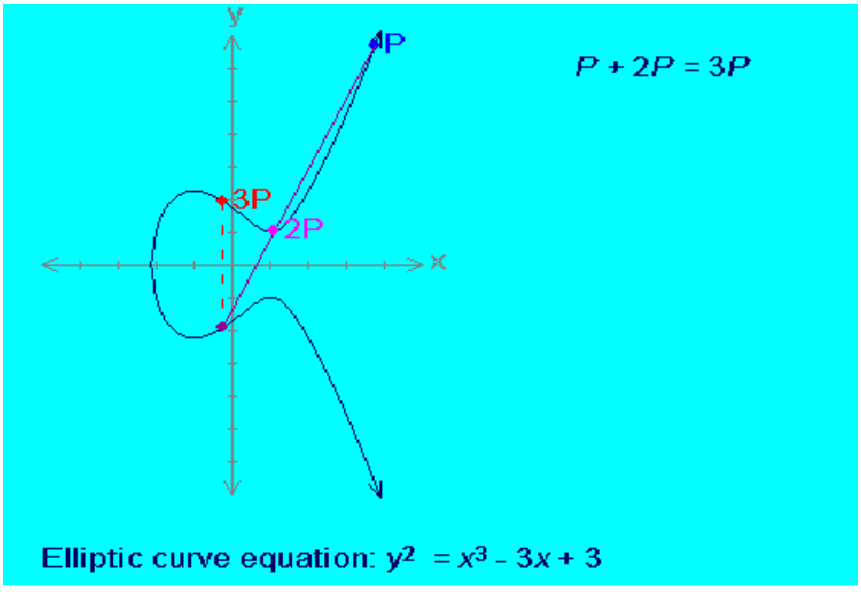
Σχήμα 22: Το σημείο στο οποίο τέμνεται η εφαπτόμενη από το P με την ελλειπτική καμπύλη και το σημείο 2P.

Έτσι προκύπτει το σημείο 2P. Με παρόμοιο τρόπο βρίσκουμε και το 3P. Φέρνουμε την ευθεία που τέμνει το 2P με το P. Η προέκτασή της τέμνει την ελλειπτική καμπύλη σε ένα τρίτο σημείο.



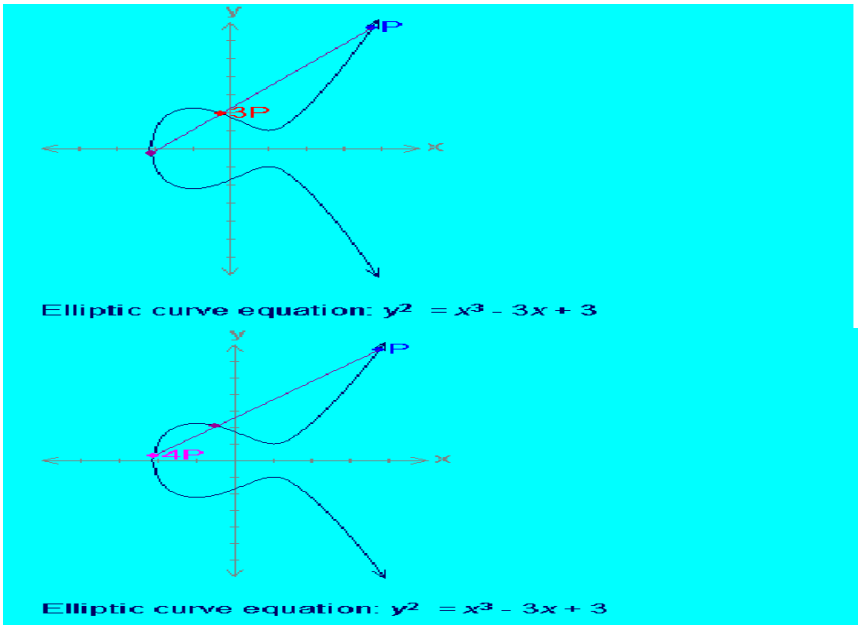
Σχήμα 23: Το σημείο στο οποίο τέμνεται το P με το σημείο 2P.

Στη συνέχεια φέρνουμε την κάθετη από αυτό το σημείο στον άξονα x. Έτσι βρίσκουμε το σημείο 3P.

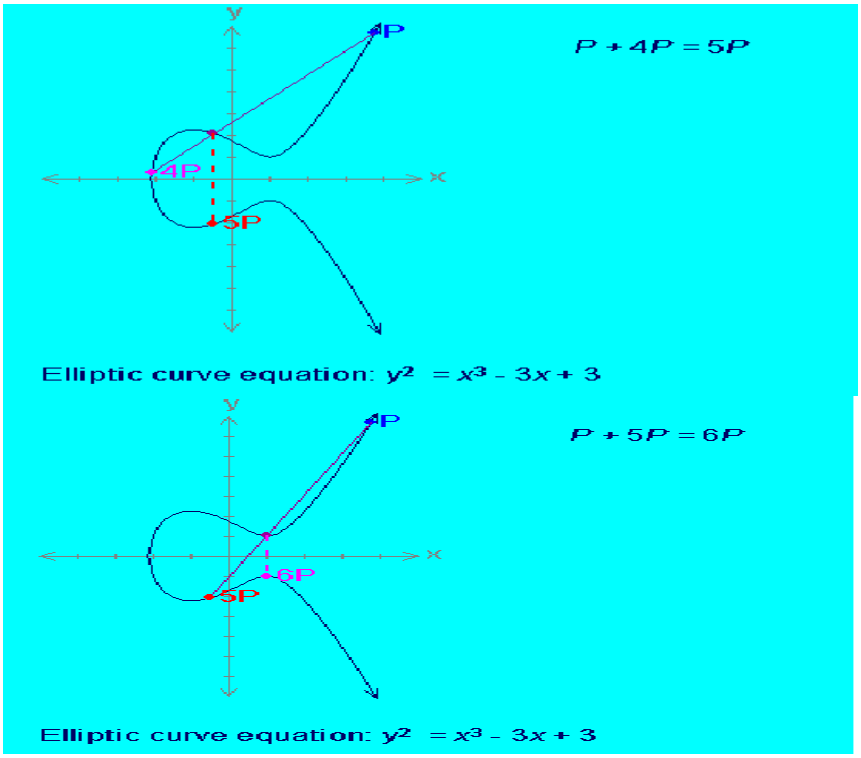


Σχήμα 24: Το σημείο 3P.

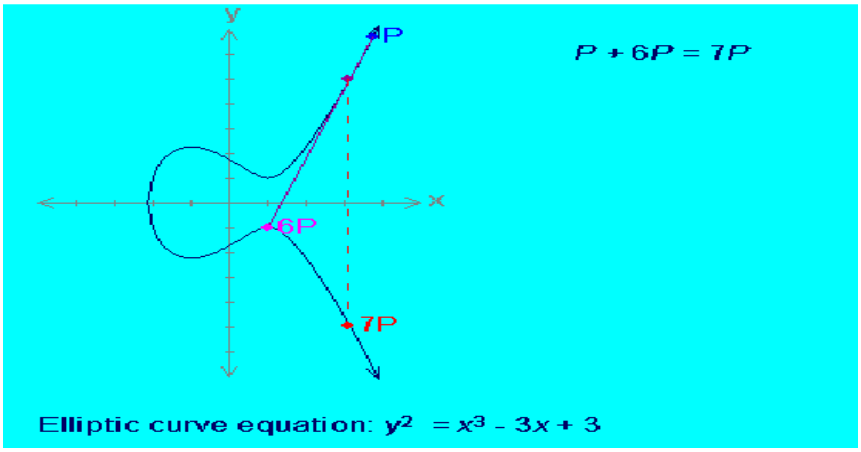
Με παρόμοιο τρόπο στα επόμενα σχήματα βρίσκουμε τα 4P, 5P, 6P και 7P αντίστοιχα.



Σχήμα 25: Το σημείο 4P.



Σχήμα 26: Το σημείο 5P και 6P.



Σχήμα 27: Το σημείο 6P και 7P.

6.12.2 Το Πρόβλημα του Διακριτού Λογαρίθμου σε Ελλειπτική Καμπύλη (ECDLP).

Στην πολλαπλασιαστική ομάδα Z_p^*, το πρόβλημα του διακριτού λογαρίθμου είναι: Δοσμένων των στοιχείων r και q της ομάδας, και p ένας πρώτος αριθμός, να βρεθεί ο αριθμός k τέτοιος ώστε r = qk mod p. Αν οι ομάδες ελλειπτικών καμπυλών περιγράφονται με τη πράξη του συμβόλου του πολλαπλασιασμού τότε το πρόβλημα του διακριτού λογαρίθμου θα είναι: Δοσμένων των σημείων P και Q σε μια ομάδα, να βρεθεί ένας αριθμός τέτοιος ώστε Pk = Q. Ο k λέγεται διακριτός λογάριθμος του Q με βάση το P. Όταν η ομάδα ελλειπτικής καμπύλης περιγράφεται από τη πράξη της πρόσθεσης, το πρόβλημα του διακριτού λογαρίθμου είναι: Δοσμένων των σημείων P και Q σε μια ομάδα να βρεθεί ο αριθμός k τέτοιος ώστε kP = Q.

Παράδειγμα 1:

Σε μια ομάδα ελλειπτικής καμπύλης της μορφής $y^2 = x^3 + 9x + 17$ στο σώμα F_{23}, ποιος είναι ο διακριτός λογάριθμος k του Q = (4, 5) με βάση το P = (16, 5);

Ένας (λογικός) τρόπος να βρούμε το k είναι να υπολογίσουμε τα πολλαπλάσια του P μέχρι να βρούμε το Q. Τα πρώτα πολλαπλάσια του P είναι:

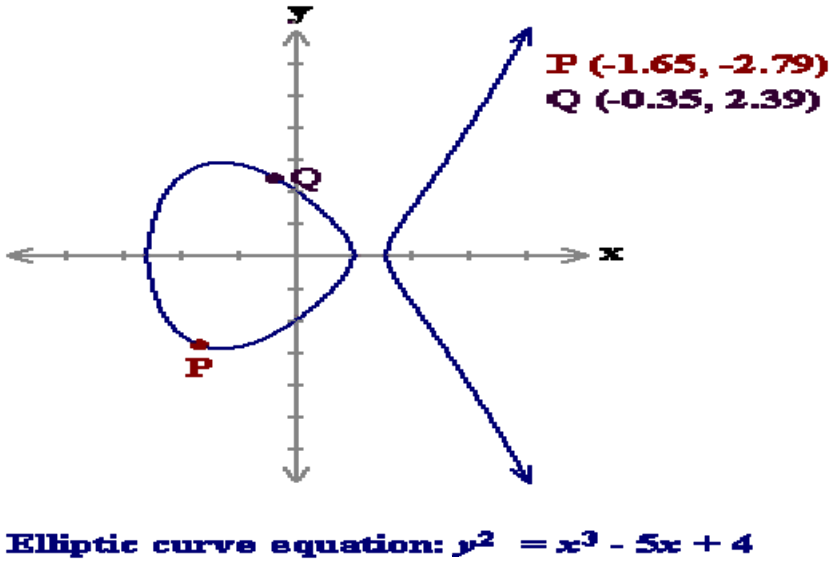
P = (16, 5), 2P = (20, 20), 3P = (14,14), 4P = (19, 20), 5P = (13,10), 6P = (7, 3), 7P = (8, 7), 8P = (12, 17), 9P = (4, 5).

Αφού το 9P = (4, 5) = Q, ο διακριτός λογάριθμος του Q με βάση το P είναι το k = 9.

Σε μια αληθινή εφαρμογή, το k θα μπορούσε να είναι αρκετά μεγάλο έτσι ώστε να είναι πολύ δύσκολο να καθοριστεί με αυτόν τον τρόπο.

Παράδειγμα 2:

Ποιος είναι ο διακριτός λογάριθμος του Q(-0,35, 2,39) με βάση το P(-1,65, -2,79) της ομάδας ελλειπτικής καμπύλης $y^2 = x^3 - 5x + 4$ των πραγματικών αριθμών;



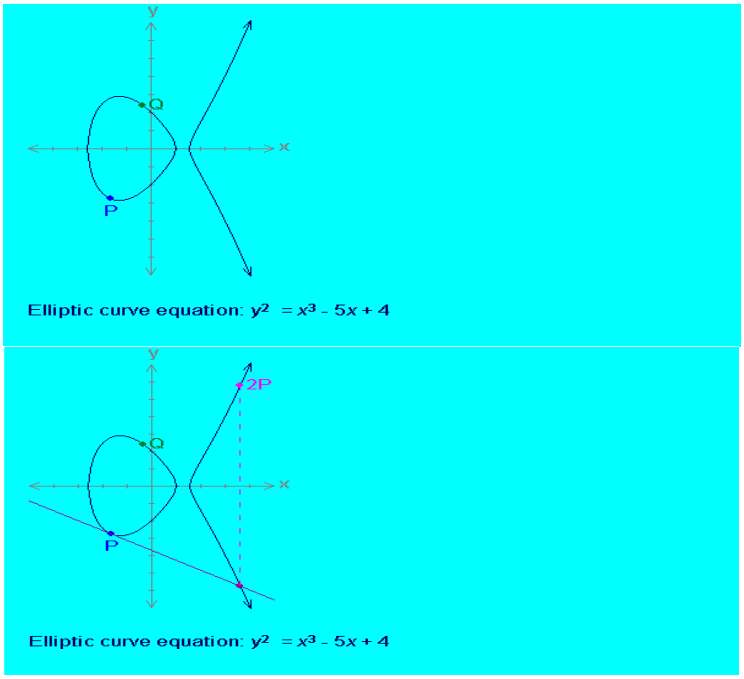
Σχήμα 28: Τα σημεία Q(-0,35, 2,39) και P(-1,65, -2,79) της ομάδας ελλειπτικής καμπύλης $y^2 = x^3 - 5x + 4$ των πραγματικών αριθμών.

6.12.3 Η λύση του προβλήματος ECDLP.

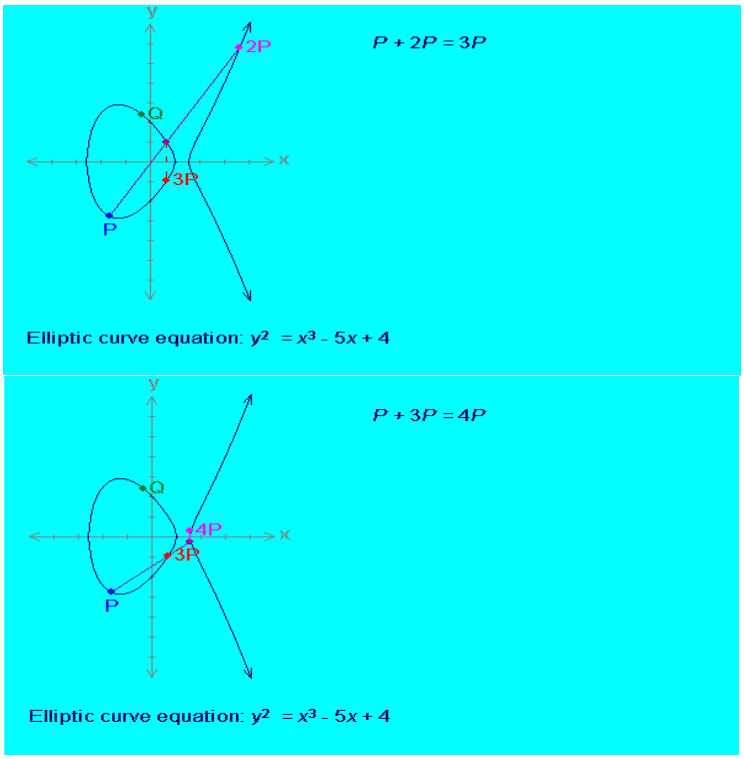
Στην ιστοσελίδα http://www.certicom.com/index.php/531-a-solution-to-the-ecdlp παρουσιάζεται ο λογάριθμος του Q με βάση το P:

Ξεκινώντας από το P βρίσκουμε 2P, 3P, 4P, 5P, 6P και 7P. Εκεί σταματάμε αφού βλέπουμε ότι ισχύει το P + 6P είναι 7P = Q. Έτσι το 7 είναι ο ζητούμενος λογάριθμος.

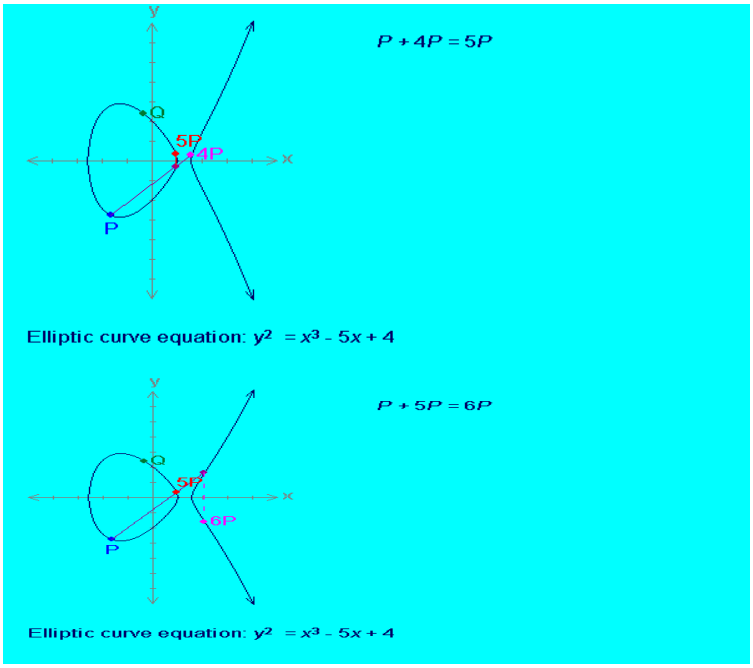
Παρακάτω ακολουθούν κάποια σχήματα που παρουσιάζουν τη λύση του προηγούμενου προβλήματος:



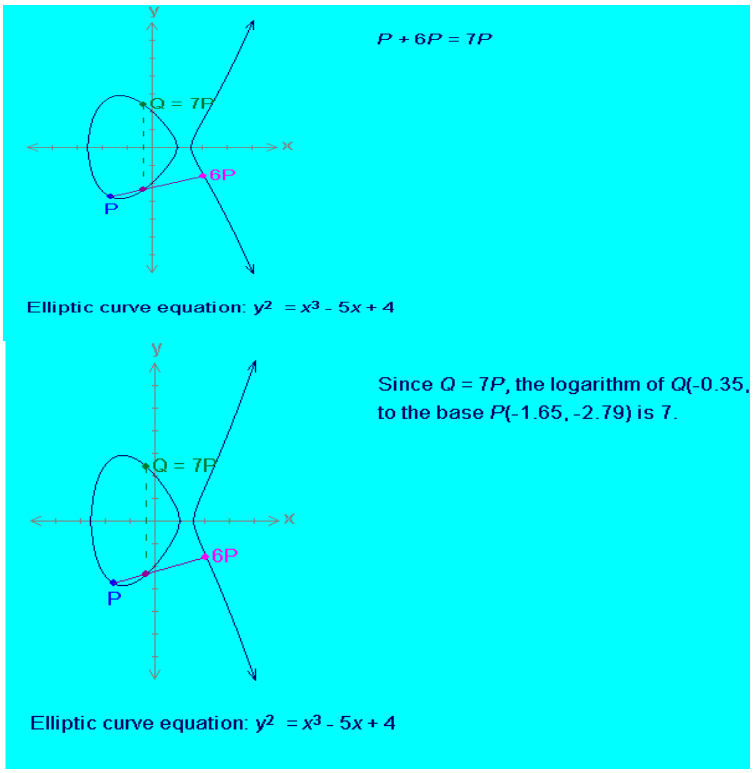
Σχήμα 29: Εύρεση του διακριτού λογάριθμου του Q(-0,35, 2,39) με βάση το P(-1,65, -2,79) της ομάδας ελλειπτικής καμπύλης $y^2 = x^3 - 5x + 4$ των πραγματικών αριθμών. Αρχικά βρίσκουμε το 2P.



Σχήμα 30: Εύρεση του 3P και 4P.



Σχήμα 31: Εύρεση του 5P και 6P.



Σχήμα 32: Εύρεση του 7P = Q.

6.13 Σύγκριση της χρήσης των ελλειπτικών καμπυλών στη λύση του προβλήματος DLP με τη χρήση συμβατικών κρυπτογραφικών συστημάτων.

Στην ενότητα αυτή συζητιούνται οι πρακτικές συνέπειες της χρήσης του συνόλου $E(F_q)$ μιας κατάλληλα επιλεγμένης ελλειπτικής καμπύλης για την εφαρμογή ενός κρυπτογραφικού συστήματος που βασίζεται στο ECDLP, αντιπαραθέτοντας την με την χρήση συμβατικών συστημάτων στο σύνολο F_p^*. Η παρατήρηση που γίνεται όσον αφορά το μέγεθος του κλειδιού είναι ότι, η καλύτερη γνωστή μέθοδος για την επίλυση του ECDLP είναι εκθετικής πολυπλοκότητας στο μέγεθος (σε bits) των $n = [\log_2 q]$ στοιχείων του σώματος F_q, ενώ αλγόριθμοι που έχουν υποεκθετική πολυπλοκότητα, για αριθμό στοιχείων $N = [\log_2 p]$, είναι διαθέσιμοι για το DLP στο F_p^* [14].

Πιο αναλυτικά, οι καλύτεροι γνωστοί γενικοί αλγόριθμοι για το ECDLP έχουν πολυπλοκότητα ανάλογη με:

$$C_{EC}(n) = 2^{n/2} = q^{1/2}$$

Ορίζεται η συνάρτηση:

$$L_p(u,c) = \exp\left(c(\log p)^u (\log\log p)^{(1-u)}\right)$$

όπου log ο φυσικός λογάριθμος. Όταν u = 1, η συνάρτηση L_p είναι εκθετική στο log p, ενώ για u = 0 είναι πολυωνυμική στο L_p. Όταν $0 < u < 1$ η συμπεριφορά της συνάρτησης είναι μεταξύ πολυωνυμικής και εκθετικής, και αναφέρεται σαν υποεκθετική.

Διακριτοί λογάριθμοι στο F_p μπορούν να βρεθούν σε χρόνο ανάλογο προς $L_p(1/3, c_0)$, όπου $c_0 = (64/9)^{1/3} \approx 1{,}92$. Σε σχέση με το N, και παραλείποντας τους σταθερούς όρους, η πολυπλοκότητα είναι:

$$C_{CONV}(N) = \exp\left(c_0 N^{1/3} (\log(N \log 2))^{2/3}\right).$$

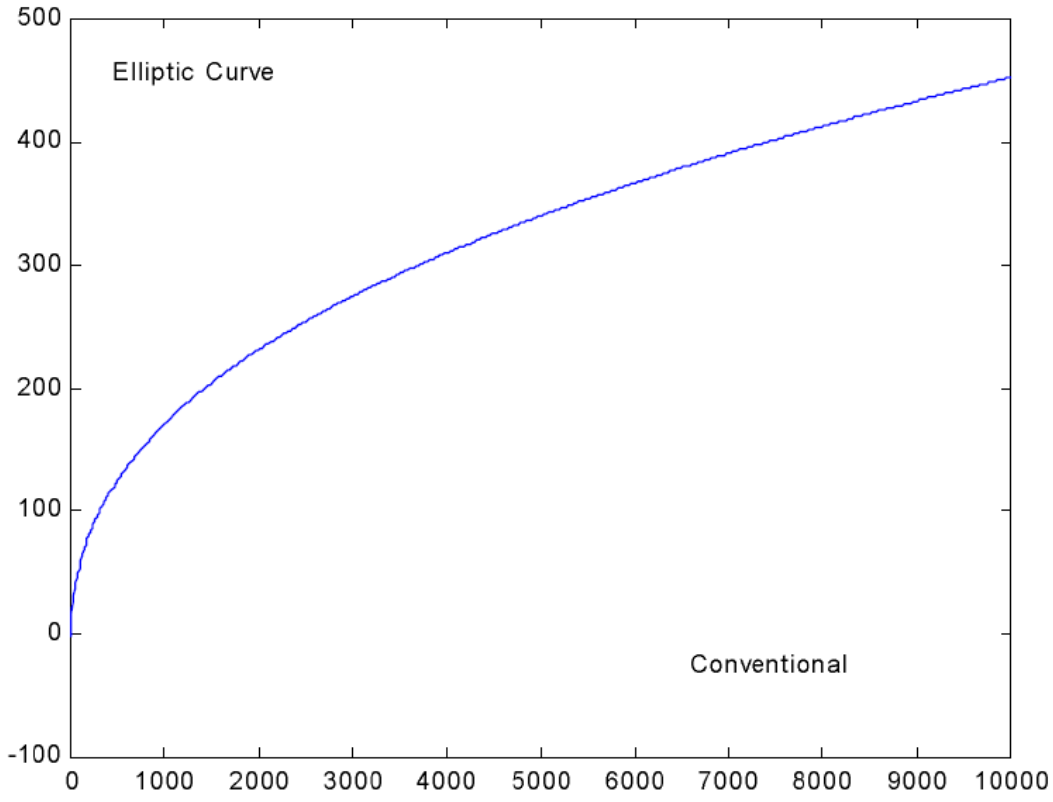
(όπου το CONV προέρχεται από το conventional = συμβατικός).

Εξισώνοντας το C_{EC} και το C_{CONV} (και παραλείποντας πάλι τους σταθερούς όρους), αποδεικνύεται ότι για παρόμοια επίπεδα ασφαλείας, πρέπει να ισχύει:

$$n = \beta N^{1/3} (\log (N \log 2))^{2/3}$$

όπου $\beta = 2c_0 / (\log 2)^{2/3} \approx 4{,}91.$

Τώρα, οι παράμετροι n και N μπορούν να ερμηνευτούν σαν τα μεγέθη των κλειδιών, σε bits, για τα αντίστοιχα κρυπτογραφικά συστήματα (ελλειπτικών καμπυλών και συμβατικών). Αυτό που παρατηρείται λοιπόν είναι ότι το μέγεθος κλειδιού σε ένα σύστημα ελλειπτικών καμπυλών αυξάνεται με ρυθμό ελαφρώς γρηγορότερο από την κυβική ρίζα του αντίστοιχου μεγέθους κλειδιού για τα συμβατικά συστήματα, για την επίτευξη περίπου ισοδύναμης κρυπτογραφικής ισχύς [15].



Σχήμα 33: Γραφική παράσταση του μεγέθους κλειδιού ενός συμβατικού. κρυπτογραφικού συστήματος σε σχέση με ένα κρυπτογραφικό σύστημα βασισμένο σε ελλειπτικές καμπύλες, ως αναφορά το πρόβλημα επίλυσης του Διακριτού Λογαρίθμου ECDLP.

Η προηγούμενη σχέση φαίνεται στη προηγούμενη γραφική παράσταση, όπου για παράδειγμα κλειδιά μεγέθους 1024 και 4096 bits (μεγέθη που χρησιμοποιούνται πολύ συχνά στον RSA) αντιστοιχούν σε κλειδιά μεγέθους 173 και 313 bits αντίστοιχα, για συστήματα ελλειπτικών καμπυλών [11].

Ωστόσο, μια δίκαιη σύγκριση θα έπρεπε να λαμβάνει επίσης υπόψη της την πολυπλοκότητα εφαρμογής του κάθε κρυπτογραφικού συστήματος. Γενικά όμως, οι βασικές πράξεις σε ένα σύστημα ελλειπτικών καμπυλών είναι πιο πολύπλοκες από τις αντίστοιχες σε ένα συμβατικό σύστημα. Παρόλα αυτά το παραπάνω σχήμα εξηγεί το πρόσφατο ενδιαφέρον για την κρυπτογραφία με τη χρήση ελλειπτικών καμπυλών αφού θεωρείται λιγότερο «ακριβή». Πρακτικά, μικρότερα μήκη κλειδιών μπορούν να μεταφραστούν σε γρηγορότερη υλοποίηση, με μικρότερη κατανάλωση ισχύος και με τη χρήση μικρότερης επιφάνειας πυριτίου.

Field size (in bits)	Size of n (in bits)	$\sqrt{\pi n / 4}$	MIPS years
163	160	2^{80}	8.5×10^{11}
191	186	2^{93}	7.0×10^{15}
239	234	2^{117}	1.2×10^{23}
359	354	2^{177}	1.3×10^{41}
431	426	2^{213}	9.2×10^{51}

Σχήμα 34: Υπολογιστική δύναμη που απαιτείται για να τον υπολογισμό του ECDLP με τη μέθοδο του Pollard.

Επίσης, αρκετά χρήσιμο για πολλά συμπεράσματα, σχετικά με την υπολογιστική δύναμη που χρειάζεται για να λυθεί το πρόβλημα του διακριτού λογαρίθμου στις ελλειπτικές καμπύλες (ECDLP) με τη μέθοδο του Pollard, είναι το σχήμα 34. Όπου MIPS, είναι Εκατομμύρια Εντολές Ανά Δευτερόλεπτο (Million Instructions Per Second). Υποθέτοντας ότι ένας υπολογιστής που έχει δύναμη 1 MIPS μπορεί να εκτελεί $4 \cdot 10^4$ προσθέσεις σημείων ελλειπτικής καμπύλης το δευτερόλεπτο, βλέπουμε τη μεγάλη δυσκολία επίλυσης του προβλήματος [1]. Επομένως, βλέπουμε ότι το ECDLP είναι αρκετά ανθεκτικό σε επιθέσεις από τρίτους.

ΚΕΦΑΛΑΙΟ 7

7. Οι κυριότερες εφαρμογές των Ελλειπτικών Καμπυλών στην Κρυπτογραφία (ECC) και οι αλγόριθμοί τους.

7.1 Εισαγωγή

Από μόνη της η Κρυπτογραφία Ελλειπτικών Καμπυλών είναι μαθηματικές εξισώσεις. Για να γίνει χρήσιμη στην ασφάλεια, αυτή πρέπει να εφαρμοστεί μέσα σε αλγόριθμους. Υπάρχουν πολλοί τέτοιοι αλγόριθμοι. Οι παρακάτω είναι οι πιο πρόσφατοι που χρησιμοποιήθηκαν στη κρυπτογραφία [8].

Ψηφιακές Υπογραφές

- ECDSA: Elliptic Curve Digital Signature Algorithm
- ECPVS: Elliptic Curve Pintsov Vanstone Signatures
- ECNR: Elliptic Curve Nyberg Rueppel

Ανταλλαγή Κλειδιού

- ECDH: Elliptic Curve Diffie - Hellman
- ECMQV: Elliptic Curve Menezes – Qu - Vanstone

Κρυπτογράφηση – Αποκρυπτογράφηση

- ECIES: Elliptic Curve Integrated Encryption Scheme

Στις επόμενες ενότητες θα γίνει περιγραφή των αντίστοιχων πρώτων από κάθε περίπτωση: ECDSA, ECDH και ECIES.

7.2 Ο αλγόριθμος δημιουργίας ψηφιακής υπογραφής με τη χρήση ελλειπτικών καμπυλών (ECDSA).

Οι ψηφιακές υπογραφές είναι το αντίστοιχο των χειρόγραφων υπογραφών σε ψηφιακή μορφή. Συγκεκριμένα πρόκειται για έναν αριθμό που εξαρτάται από μία μυστική πληροφορία (μυστικό κλειδί) που κατέχει μόνο ο υπογράφων και από το μήνυμα που υπογράφεται. Η ψηφιακή υπογραφή θα πρέπει να είναι επαληθεύσιμη, δηλαδή να μπορεί να αποδειχθεί ότι όντως ανήκει στο άτομο που υπέγραψε χωρίς να χρειάζεται να αποκαλυφθεί το μυστικό κλειδί [16].

Μία κατηγορία ψηφιακών υπογραφών που χρησιμοποιούνται είναι εκείνες που κατασκευάζονται με τη χρήση ελλειπτικών καμπυλών [17]. Όπως γνωρίζουμε η κρυπτογράφηση γίνεται με τη βοήθεια του δημοσίου κλειδιού και η αποκρυπτογράφηση γίνεται μόνον από αυτόν που γνωρίζει το ιδιωτικό κλειδί. Αντίθετα, στις ψηφιακές υπογραφές μόνο αυτός που υπογράφει έχει το μυστικό κλειδί και αυτός που επικυρώνει την υπογραφή πρέπει να έχει το δημόσιο.

Στην ενότητα αυτή θα περιγραφεί ο αλγόριθμος για κατασκευή ψηφιακής υπογραφής με τη χρήση ελλειπτικών καμπυλών (Elliptic Curve Digital Signature Algorithm - ECDSA), όπως αναφέρεται στην εργασία [18] και στο βιβλίο [19].

Ο ECDSA αλγόριθμος έχει να κάνει με μη συμμετρικές υπογραφές με παράρτημα (asymmetric digital signature with appendix). Μη συμμετρική είναι η ψηφιακή υπογραφή που χρειάζεται ένα ιδιωτικό κλειδί για να δημιουργηθεί και ένα αντίστοιχο δημόσιο για να επαληθευτεί. Υπογραφή με παράρτημα είναι αυτή που δεν εφαρμόζεται απευθείας στο μήνυμα αλλά στην έξοδο μιας hash συνάρτησης που έχει σαν είσοδο το μήνυμα αυτό. Όπως θα δούμε παρακάτω εφαρμόζοντας τον αλγόριθμο, η ασφάλεια των ψηφιακών υπογραφών βασίζεται στην δυσκολία επίλυσης του προβλήματος του διακριτού λογαρίθμου (ECDLP) [15].

7.2.1 Παράμετροι σώματος.

Οι παράμετροι σώματος (domain parameters) για ένα κρυπτογραφικό πρωτόκολλο (βασισμένο στις ελλειπτικές καμπύλες) αποτελούνται από μία κατάλληλα επιλεγμένη ελλειπτική καμπύλη E ορισμένη σε ένα πεπερασμένο σώμα F_q και ένα σημείο $P \in E(F_q)$. Οι παράμετροι σώματος είτε χρησιμοποιούνται από ένα μόνο χρήστη, είτε από μία ομάδα από χρήστες.

Πιο συγκεκριμένα, οι παράμετροι σώματος είναι οι παρακάτω (στη συγκεκριμένη περίπτωση οι παράμετροι σώματος είναι σχετικοί με αυτούς ενός δυαδικού σώματος).

1. Το μέγεθος του πεπερασμένου σώματος, $q= 2^m$.
2. Μία ένδειξη FR (Field Representation) που δείχνει αν τα στοιχεία εκφράζονται σε πολυωνυμική ή κανονική βάση.
3. (Προαιρετικά) Μία συμβολοσειρά τουλάχιστον 160 ψηφίων η οποία χρησιμεύει στο να αποδείξει ότι η ελλειπτική καμπύλη έχει προκύψει τυχαία.
4. Δύο στοιχεία $a_2, a_6 \in F_q$ τα οποία εκράζουν την ελλειπτική καμπύλη E πάνω στο F_q.
5. Δύο στοιχεία $x_P, y_P \in F_q$, τα οποία ορίζουν ένα πεπερασμένο σημείο $P = (x_P, y_P)$ πάνω στην E, του οποίου η τάξη είναι πρώτος αριθμός.
6. Η τάξη r του P, όπου $r > 2^{160}$ και $r > 4\sqrt{q}$.
7. Ο αριθμός k για τον οποίο ισχύει $k = N/n$, όπου N η τάξη της καμπύλης.

7.2.2 Δημιουργία κλειδιού.

Έστω ότι έχει κατασκευαστεί ένα σύνολο από παραμέτρους σώματος όπως ορίστηκαν παραπάνω. Ένας χρήστης A επιλέγει έναν ακέραιο d, όπου $0< d < r$ και υπολογίζει το σημείο $Q= dP$. Το σημείο Q είναι το δημόσιο κλειδί του A και το d είναι το κρυφό κλειδί.

7.2.3 Δημιουργία ψηφιακής υπογραφής.

Για να υπογράψει ένα μήνυμα m, ο χρήστης Α εκτελεί τα ακόλουθα βήματα:

1. Διαλέγει ένα τυχαίο ακέραιο n, όπου $0 < n < r$.
2. Υπολογίζει $nQ = (x_1, y_1)$ και $t = x_1 \bmod r$. Αν $t = 0$ τότε επιστρέφει στο βήμα 1.
3. Υπολογίζει το $n^{-1} \bmod r$.
4. Υπολογίζει το $e = hash(m)$, δηλαδή εφαρμόζει μία συγκεκριμένη συνάρτηση κατακερματισμού στο μήνυμα m.
5. Υπολογίζει το $s = n^{-1}(e + dt) \bmod r$. Αν $s = 0$, επιστρέφει στο βήμα 1.
6. Η υπογραφή του Α για το μήνυμα m είναι το ζευγάρι (t, s).

Σημειώνεται ότι η συνάρτηση κατακερματισμού (hash function) που αναφέρεται στο βήμα 4 είναι ουσιαστικά μια αντιστοίχηση ένα – προς – πολλά η οποία αντιστοιχεί το μήνυμα, που μπορεί να είναι αρκετά εκτενές, σε έναν αριθμό περιορισμένου μεγέθους με τέτοιο τρόπο ώστε να είναι πρακτικά αδύνατο να πραγματοποιηθεί η αντίστροφη αντιστοίχιση. Ο αριθμός που προκύπτει με την συνάρτηση κατακερματισμού είναι αυτός που χρησιμοποιείται κατά τους υπολογισμούς. Στο παράρτημα Ε παρουσιάζεται ο κώδικας σε Java για την υλοποίηση και την επικύρωση μιας ψηφιακής υπογραφής.

7.2.4 Επικύρωση ψηφιακής υπογραφής.

Αν ο χρήστης Β θέλει να εξακριβώσει ότι η υπογραφή (t, s) που συνοδεύει το μήνυμα m ανήκει πράγματι στο χρήστη Α, εκτελεί τα ακόλουθα βήματα:

1. Αποκτά ένα επικυρωμένο αντίγραφο των παραμέτρων σώματος και του δημοσίου κλειδιού του Α.
2. Επαληθεύει ότι οι αριθμοί t και s ανήκουν στο διάστημα [1, r - 1].
3. Υπολογίζει το $e = hash(m)$.
4. Υπολογίζει το $w = s^{-1} \bmod r$.
5. Υπολογίζει το $u_1 = ew \bmod r$ και $u_2 = tw \bmod r$.

6. Υπολογίζει το $u_1P + u_2Q = (x_1, y_1)$. Αν $x_1 = x_2 = 0$, απορρίπτει την υπογραφή, αλλιώς υπολογίζει το $\upsilon = x_1 \bmod r$.
7. Αποδέχεται την υπογραφή, αν και μόνο αν $\upsilon = t$.

Το ότι ο αλγόριθμος επικύρωσης είναι σωστός αποδεικνύεται ως εξής:

Αν η υπογραφή (t, s) έχει προέλθει πράγματι από τον A, τότε θα ισχύει $s = n^{-1}(e + dt) \equiv s^{-1}e + s^{-1}dt \equiv we + wtd \equiv u_1 + u_2 d \pmod r$.

Άρα, $u_1P + u_2Q = (u_1 + u_2 d)P = nP$, επομένως το υ θα ταυτίζεται με το t.

7.3 Ο αλγόριθμος ανταλλαγής κλειδιού με τη χρήση ελλειπτικών καμπυλών των Diffie-Hellman (ECDH).

Οι Diffie και Hellman ανέπτυξαν το key agreement protocol, το οποίο ονομάζεται επίσης και exponential key agreement protocol, το 1976. Το δημοσίευσαν για πρώτη φορά στο περιοδικό New Grounds in Cryptography. Ουσιαστικά, το πρωτόκολλο αυτό επιτρέπει σε δύο χρήστες να ανταλλάξουν ένα μυστικό κλειδί με ασφάλεια εντός ενός μη ασφαλούς μέσου [20].

7.3.1 Παράμετροι σώματος.

Το πρωτόκολλο έχει δύο παραμέτρους του συστήματος, τα p και g.

Είναι και οι δύο δημόσιες παράμετροι και μπορούν να χρησιμοποιηθούν από όλους τους χρήστες του συστήματος.

- Η παράμετρος p: είναι ένας πρώτος αριθμός.
- Η παράμετρος g: ονομάζεται συνήθως γεννήτορας (generator) και είναι ένας ακέραιος μικρότερος του p που μπορεί να παράγει κάθε στοιχείο από το 1 έως το p-1 όταν πολλαπλασιάζει τον εαυτό του, έναν συγκεκριμένο αριθμό φορών, με modulo τον πρώτο αριθμό p.

$$\alpha^0, \alpha^1, \ldots\ldots\ldots\ldots, \alpha^{p-2}$$

Σχήμα 1: Ο α είναι γεννήτορας των α^0, α^1,…., α^{p-2}.

7.3.2 Δημιουργία μυστικού κλειδιού.

Ας υποθέσουμε ότι έχουμε τον Α και τον Β, οι οποίοι θέλουν να συμφωνήσουν στο να έχουν από κοινού ένα μυστικό κλειδί σύμφωνα με το πρωτόκολλο των Diffie & Hellman προκειμένου να ανταλλάσσουν μηνύματα.

Ο Α δημιουργεί ένα σπάνιο μυστικό αριθμό α, που είναι το ιδιωτικό του κλειδί και ο Β δημιουργεί παρόμοια ένα σπάνιο αριθμό b.

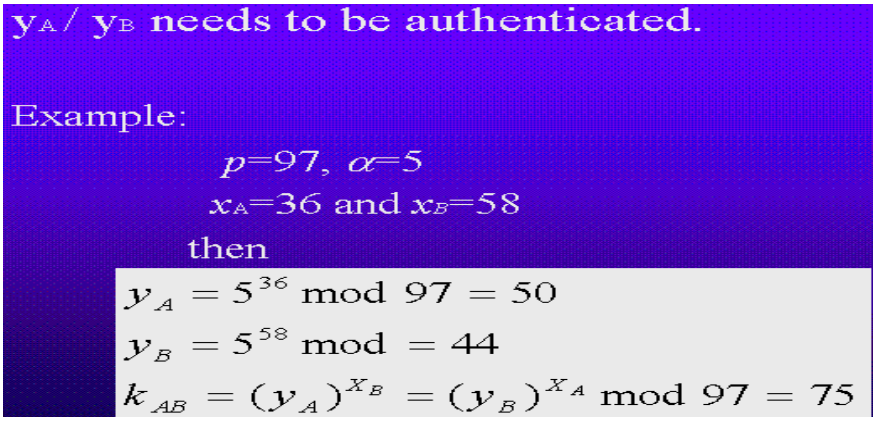
Έπειτα δημιουργούν τα δημόσια κλειδιά τους σε συνδυασμό με τα p, g και με τα ιδιωτικά τους κλειδιά.

Το δημόσιο κλειδί του Α είναι το $g^a \bmod p$ και του Β το $g^b \bmod p$. Έπειτα ανταλλάσσουν τα δημόσια κλειδιά τους.

Ο Α πληκτρολογεί $K_{ab} = (g^b)^a \bmod p$ και ο Β παρόμοια $K_{ba} = (g^a)^b \bmod p$. Εφόσον $K_{αb} = K_{bα} = K$, ο Α και ο Β έχουν τώρα μαζί ένα μυστικό κλειδί Κ.

Το πρωτόκολλο βασίζεται, ως αναφορά την ασφάλειά του στο πρόβλημα του διακριτού λογαρίθμου. Θεωρεί ότι είναι υπολογιστικά πολύ δύσκολο να υπολογίσει κανείς το μυστικό κλειδί $K = g^{ab} \bmod p$ κι ας γνωρίζει τις δύο δημόσιες μεταβλητές $g^a \bmod p$ και $g^b \bmod p$, γιατί ο πρώτος αριθμός p είναι πάρα πολύ μεγάλος.

Στο σχήμα 36 φαίνεται ένα παράδειγμα εφαρμογής του πρωτοκόλλου Diffie – Hellman. Ο α είναι ο γεννήτορας, ο p είναι ο πρώτος αριθμός, ο x_A είναι το ιδιωτικό κλειδί του Α και ο x_B είναι το ιδιωτικό κλειδί του Β.



Σχήμα 2: Εφαρμογή της δημιουργίας δημόσιου κλειδιού με το πρωτόκολλο Diffie – Hellman.

7.3.3 Χρήση των ελλειπτικών καμπυλών στην ανταλλαγή κλειδιών (ECDH).

Στην ανταλλαγή κλειδιών του αλγόριθμου Diffie – Hellman με τη χρήση ελλειπτικών καμπυλών, οι δύο μονάδες που επικοινωνούν ο S και ο C συμφωνούν προκαταβολικά να χρησιμοποιήσουν την ίδια ελλειπτική καμπύλη με τις ίδιες παραμέτρους και το ίδιο στοιχείο βάσης G. Έτσι, ο καθένας τους δημιουργεί το ιδιωτικό του κλειδί Pr_s και Pr_c, αντίστοιχα. Έτσι, το δημόσιο κλειδί που προκύπτει θα είναι $Pu_s = Pr_s \cdot G$ και $Pu_c = Pr_c \cdot G$.

Στη συνέχεια και οι δύο ανταλλάσσουν τα δημοσιά τους κλειδιά, και ο καθένας πολλαπλασιάζει το ιδιωτικό του κλειδί με το δημόσιο κλειδί του άλλου για να πάρουν το κοινό δημόσιο κλειδί το $Pr_c \cdot Pu_s = Pr_s \cdot Pu_c = Pr_s \cdot Pr_c \cdot G$. Ο επιτιθέμενος δε μπορεί να καθορίσει αυτό το δημόσιο κλειδί από τις παραμέτρους της ελλειπτικής καμπύλης, από το G, αλλά ούτε και από το δημόσιο κλειδί του καθενός προσώπου χωριστά [21], [22].

7.3.4 Προσπάθεια υποκλοπής του κλειδιού από κάποιο τρίτο πρόσωπο.

Έχουμε ένα τρίτο άτομο, τον Γ, ο οποίος υποκλέπτει το δημόσιο κλειδί του Α και δίνει το δικό του δημόσιο κλειδί στον Β. Το ίδιο πράττει και με τον Β αντιστρόφως. Έτσι λοιπόν ο Γ καταλήγει να συμφωνεί με τον Α και Β χωριστά για

να μοιραστούν το μυστικό κλειδί. Μετά από αυτή την ανταλλαγή και έπειτα από την προαναφερθείσα διαδικασία, ο Γ μπορεί πλέον να αποκωδικοποιεί τα μηνύματα που στέλνει ο Α στο Β, να τα διαβάζει και συχνά να τα μεταποιεί όταν τα επανακρυπτογραφεί με το σωστό κλειδί για να τα στείλει στο σωστό παραλήπτη, δηλαδή στον Β. Το ίδιο γίνεται και αντίστροφα.

Η αδυναμία αυτή του πρωτοκόλλου σε τέτοιες επιθέσεις παρουσιάζεται διότι μέσα στη διαδικασία ανταλλαγής κλειδιών δεν εξασφαλίζεται πουθενά η αυθεντικότητα των δύο μερών που συναλλάσσονται. Μια καλή λύση αντιμετώπισης αυτού του προβλήματος είναι η πιστοποίηση της αυθεντικότητας μέσω ψηφιακών πιστοποιητικών και υπογραφών.

7.4 Ο ολοκληρωμένος αλγόριθμος κρυπτογράφησης – αποκρυπτογράφησης με τη χρήση ελλειπτικών καμπυλών (ECIES).

Ο ολοκληρωμένος αλγόριθμος κρυπτογράφησης είναι ένας αλγόριθμος κρυπτογράφησης δημοσίου κλειδιού ο οποίος παρέχει σημαντική ασφάλεια απέναντι σε εχθρικές προσπάθειες για υποκλοπή μηνυμάτων. Δύο υλοποιήσεις του αλγορίθμου υπάρχουν [23]:

- DLIES: Discrete Logarithm Integrated Encryption Scheme
- ECIES: Elliptic Curve Integrated Encryption Scheme

Ο αλγόριθμος ECIES είναι και ο τελευταίος που υλοποιήθηκε και ο πιο σημαντικός.

7.4.1 Παράμετροι για την κρυπτογράφηση του μηνύματος.

Για να στείλει η Alice ένα κρυπτογραφημένο μήνυμα στον Bob με το ECIES χρειάζεται τις παρακάτω πληροφορίες [23], [24], [25], [26]:

- Το κρυπτογραφικό πακέτο που θα χρησιμοποιηθεί:
    - Η συνάρτηση δημιουργίας κλειδιού KDF, φτιαγμένη με το πρότυπο ANSI-X9.63.

- Ο κώδικας πιστοποίησης του μηνύματος MAC, φτιαγμένος με το πρότυπο, HMAC-SHA-1-160 με 160-bit μήκος κλειδιού.
- Ο αλγόριθμος συμμετρικού κλειδιού, symmetric encryption scheme E, φτιαγμένος με το πρότυπο, 3-key TDES με τη μέθοδο CBC ή το κρυπτογραφικό σχήμα XOR.

- Οι παράμετροι ελλειπτικής καμπύλης: (p, a, b, G, n, h) για μια ελλειπτική καμπύλη σε ένα σώμα πρώτων αριθμών ή (m, f(x), a, b, G, n, h) για μια ελλειπτική καμπύλη σε ένα δυαδικό σώμα.
- Το δημόσιο κλειδί του Bob: K_B (Ο Bob δημιουργεί το κλειδί ως εξής: $K_B = k_B G$, όπου k_B είναι το ιδιωτικό κλειδί το οποίο διαλέγει τυχαία: $k_B \in [1, n-1]$).
- Προαιρετική πληροφορία προς διανομή: S_1 και S_2.

7.4.2 Αλγόριθμος κρυπτογράφησης του μηνύματος.

Για να κρυπτογραφήσει το μήνυμα m η Alice κάνει τα παρακάτω:

1. Δημιουργεί ένα τυχαίο αριθμό $r \in [1, n-1]$ και υπολογίζει το $R = rG$.
2. Παράγει ένα κοινό μυστικό : $S = P_x$, όπου $P = (P_x, P_y) = rK_B$ (και $P \neq O$).
3. Χρησιμοποιεί τη συνάρτηση KDF για να παράγει συμμετρική κρυπτογράφηση και τον MAC για να καθορίσει το μήκος των κλειδιών: $k_E \| k_M = \text{KDF}(S \| S_1)$.
4. Κρυπτογραφεί το μήνυμα: $c = E(k_E; m)$.
5. Υπολογίζει την ετικέτα του κρυπτογραφημένου μηνύματος και το S_2: $d = \text{MAC}(k_M; c \| S_2)$.
6. Παράγει το $R \| c \| d$.

7.4.3 Αλγόριθμος αποκρυπτογράφησης του μηνύματος.

Για να αποκρυπτογραφήσει το κρυπτογραφημένο μήνυμα R || c || d, ο Bob εκτελεί τα παρακάτω:

1. Παράγει το κοινό μυστικό: $S = P_x$, όπου $P = (P_x, P_y) = k_B R$ (είναι το ίδιο που παράχθηκε από την Alice διότι $P = k_B R = k_B rG = rk_B G = rK_B$), ή η έξοδος αποτυγχάνει αν P = O.
2. Παράγει τα κλειδιά με τον ίδιο τρόπο όπως η Alice: $k_E \| k_M = \mathrm{KDF}(S \| S_1)$.
3. Χρησιμοποιεί τον MAC για να ελέγξει την ετικέτα και εξάγει το μήνυμα «αποτυχία» αν $d \neq \mathrm{MAC}(k_M ; c \| S_2)$.
4. Χρησιμοποιεί το συμμετρικό αλγόριθμο αποκρυπτογράφησης για να αποκρυπτογραφήσει το μήνυμα $m = E^{-1}(k_E; c)$.

ΚΕΦΑΛΑΙΟ 8

8. Συμπεράσματα

Ο στόχος της παρούσας πτυχιακής εργασίας ήταν η παρουσίαση και ανάλυση της κρυπτογραφίας ως μια επιστήμη που χρησιμοποιήθηκε για να εξυπηρετήσει την πληροφορική. Η τέχνη της κρυπτογραφίας προϋπήρχε πολύ πριν την ανάπτυξη της επιστήμης της πληροφορικής. Για αυτό το λόγο έγινε μια ιστορική αναδρομή και εξηγήθηκε η εμφάνιση και η χρήση της κρυπτογραφίας μέσα στο χρόνο. Η κρυπτογραφία ξεκίνησε ως τέχνη και μεταμορφώθηκε σε επιστήμη και η χρήση της στην πληροφορική καθίσταται αναγκαία πια.

Είναι φανερό ότι ένα αρκετά σημαντικό μαθηματικό υπόβαθρο χρειάζεται για τη σωστή και επαρκή κατανόηση των σημαντικότερων θεωριών που διέπουν τη λειτουργία των περισσότερων αλγόριθμων που χρησιμοποιούνται στη κρυπτογραφία. Στην παρούσα εργασία έγινε προσπάθεια για την καλύτερη δυνατή εξήγηση των θεωριών και των βασικών αρχών των μαθηματικών που διέπουν αυτούς τους αλγορίθμους. Τα μαθηματικά που χρησιμοποιήσαμε εξηγήθηκαν και αναλύθηκαν με όσο το δυνατό πιο απλό και κατανοητό τρόπο. Αυτό έγινε και με τη παρουσίαση απλών παραδειγμάτων προς λύση και εξάσκηση.

Με την ανάλυση των σπουδαιότερων αλγορίθμων τόσο της συμμετρικής, αλλά και της ασύμμετρης κρυπτογραφίας είδαμε ότι και οι δύο τεχνικές έχουν η καθεμιά τους τα πλεονεκτήματα και τα μειονεκτήματα τους ανάλογα με τη χρήση τους. Η συμμετρική κρυπτογραφία με αλγορίθμους, όπως ο DES και IDEA παλιότερα, αλλά και ο AES, ως πιο καινούριος, είναι πολύ πιο γρήγορη αν και όχι τόσο ασφαλής και το κόστος της είναι αρκετά μικρό σε υλοποίηση. Αντίθετα, η ασύμμετρη κρυπτογραφία με αλγορίθμους, όπως ο RSA και ο ElGamal, χρησιμοποιείται πιο εύκολα στα σημερινά υπολογιστικά συστήματα που έχουν να κάνουν με τις επικοινωνίες στην εποχής μας. Οι αλγόριθμοι αυτοί βέβαια είναι πιο αργοί αλλά η ασφάλεια τους είναι πολύ μεγαλύτερη.

Η αναδρομή μας τελικά ολοκληρώνεται με τη παρουσίαση της κρυπτογραφίας με τη χρήση των ελλειπτικών καμπυλών. Προσπαθήσαμε να

εξηγήσουμε τη χρήση των ελλειπτικών καμπυλών στη κρυπτογραφία με απλά παραδείγματα των σημαντικότερων αλγορίθμων των ελλειπτικών καμπυλών. Με τη βοήθεια αυτών των εφαρμογών τεκμηριώθηκε και πρακτικά η χρησιμότητα των ελλειπτικών καμπυλών στην κρυπτογραφία.

Αυτό που πιστεύουμε ότι πρέπει να τονιστεί είναι τα αρκετά μικρότερα μεγέθη κλειδιών που χρησιμοποιούνται σε κρυπτοσυστήματα ελλειπτικών καμπυλών σε σχέση με τα αντίστοιχα μεγέθη κλειδιών παλιότερων συμβατικών κρυπτοσυστημάτων, για να πετύχουμε το ίδιο επίπεδο ασφάλειας. Επίσης, βλέπουμε ότι όλο και περισσότεροι επιστήμονες ασχολούνται με την κρυπτογραφία και τη χρήση ελλειπτικών καμπυλών επάνω σε αυτή. Πολλές εταιρείες και οργανισμοί ασχολούνται επάνω σε αυτόν τον τομέα.

Σαν παράδειγμα αναφέρουμε τη Certicom, της οποίας η ιστοσελίδα είναι αρκετά πρωτοποριακή. Το αντίστοιχο εκπαιδευτικό εγχειρίδιο (tutorial), σχετικά με τις ελλειπτικές καμπύλες, που βρίσκεται στην ιστοσελίδα της είναι αρκετά χρήσιμο. Η Certicom, διοργανώνοντας συχνά σεμινάρια και διαγωνισμούς, έχει βάλει για τα καλά τις ελλειπτικές καμπύλες στη κρυπτογραφία και τις θεωρεί απαραίτητο κομμάτι για την υλοποίηση των σπουδαιότερων κρυπτογραφικών συστημάτων. Έτσι, περαιτέρω έρευνα και υλοποίηση νέων αλγορίθμων με τη χρήση των ελλειπτικών καμπυλών θα ήταν πολύ χρήσιμη για την κρυπτογραφία.

Παράρτημα

Α. Κώδικας υλοποίησης του Αλγόριθμου του Καίσαρα σε java.

Το πρόγραμμα διαβάζει το μήνυμα εισόδου και το εξάγει σε μια καθορισμένη έξοδο. Το κλειδί είναι ένας ακέραιος που δίνουμε εμείς από το 0-25. Το πρόγραμμα κάνει και κρυπτογράφηση και αποκρυπτογράφηση μετακινώντας τους χαρακτήρες τόσες θέσεις όσες είναι το κλειδί key με τη μέθοδο rotate.

```
// Caesar.java: implement the Caesar cipher
//   This carries out a simple rotation of lower-case letters,
//   and does nothing to all other characters, making the decryption
//   process even easier, because caps and punctuation marks survive
unchanged.
//   Usage: java Caesar (-d | -e) key
//   Above, option "-d" is for decryption, while "-e" is for
encryption
import java.io.*;
public class Caesar {
   private Reader in; // standard input stream for message
   private int key; // (en|de)cryption key

   // Caesar: constructor, opens standard input, passes key
   public Caesar(int k) {
      // open file
      in = new InputStreamReader(System.in);
      key = k;
   }
   // (en|de)crypt: just feed in opposite parameters
   public void encrypt() { translate(key); }
   public void decrypt() { translate(-key); }

   // translate: input message, translate
   private void translate(int k) {
      char c;
      while ((byte)(c = getNextChar()) != -1) {
         if (Character.isLowerCase(c)) {
            c = rotate(c, k);
         }
         System.out.print(c);
      }
   }

   // getNextChar: fetches next char.
   public char getNextChar() {
      char ch = ' '; // = ' ' to keep compiler happy
      try {
         ch = (char)in.read();
      } catch (IOException e) {
         System.out.println("Exception reading character");
```

```
      }
      return ch;
   }
   // rotate: translate using rotation, version with table lookup
   public char rotate(char c, int key) { // c must be lowercase
      String s = "abcdefghijklmnopqrstuvwxyz";
      int i = 0;
      while (i < 26) {
          // extra +26 below because key might be negative
          if (c == s.charAt(i)) return s.charAt((i + key + 26)%26);
             i++;
      }
      return c;
   }


   // main: check command, (en|de)crypt, feed in key value
   public static void main(String[] args) {
      if (args.length != 2) {
         System.out.println("Usage: java Caesar (-d | -e) key");
         System.exit(1);
      }
      Caesar cipher = new Caesar(Integer.parseInt(args[1]));
      if (args[0].equals("-e")) cipher.encrypt();
      else if (args[0].equals("-d")) cipher.decrypt();
      else {
         System.out.println("Usage: java Caesar (-d | -e) key");
         System.exit(1);
      }
   }
}
```

Εδώ παρουσιάζονται τα αποτελέσματα μετά την εκτέλεση του προγράμματος. Το αρχείο message.txt περιέχει το κείμενο «a quotation for Ecclesiastes». Το κλειδί που χρησιμοποιούμε είναι το 3 και γράφουμε –e για κρυπτογράφηση και –d για αποκρυπτογράφηση:

```
% cat message.text
i returned, and saw under the sun, that the race is not to the
swift,
nor the battle to the strong, neither yet bread to the wise, nor yet
riches
to men of understanding, nor yet favour to men of skill; but time
and chance
happeneth to them all.

% java Caesar -e 3 < message.text
l uhwxuqhg, dqg vdz xqghu wkh vxq, wkdw wkh udfh lv qrw wr wkh
vzliw,
qru wkh edwwoh wr wkh vwurqj, qhlwkhu bhw euhdg wr wkh zlvh, qru bhw
ulfkhv
wr phq ri xqghuvwdqglqj, qru bhw idyrxu wr phq ri vnloo; exw wlph
dqg fkdqfh
kdsshqhwk wr wkhp doo.

% java Caesar -e 3 < message.text | java Caesar -d 3
i returned, and saw under the sun, that the race is not to the
swift,
```

```
nor the battle to the strong, neither yet bread to the wise, nor yet
riches
to men of understanding, nor yet favour to men of skill; but time
and chance
happeneth to them all.
```

Β. Ψευδοκώδικας παραγωγής του Κλειδιού γύρου ή Expansion Key για τον αλγόριθμο AES.

```
KeyExpansion(byte key[4*Nk], word w[Nb*(Nr+1)], Nk)
begin
   word  temp

   i = 0

   while (i < Nk)
      w[i] = word(key[4*i], key[4*i+1], key[4*i+2], key[4*i+3])
      i = i+1
   end while

   i = Nk

   while (i < Nb * (Nr+1)]
      temp = w[i-1]
      if (i mod Nk = 0)
         temp = SubWord(RotWord(temp)) xor Rcon[i/Nk]
      else if (Nk > 6 and i mod Nk = 4)
         temp = SubWord(temp)
      end if
      w[i] = w[i-Nk] xor temp
      i = i + 1
   end while
end

Note that Nk=4, 6, and 8 do not all have to be implemented;
they are all included in the conditional statement above for
conciseness.   Specific implementation requirements for the
Cipher Key are presented in Sec. 6.1.
```

Γ. Αλγόριθμος του Ευκλείδη.

Στα Στοιχεία του Ευκλείδη το Βιβλίο VII αρχίζει με τον αλγόριθμο του Ευκλείδη, με τον οποίο βρίσκουμε το μέγιστο κοινό διαιρέτη (ΜΚΔ) δύο αριθμών.

Είναι ένας από τους παλαιότερους αλγόριθμους με μεγάλη σπουδαιότητα, καθώς για την εύρεση του ΜΚΔ δεν απαιτείται παραγοντοποίηση των ακεραίων.

```
GCD(a, b)
1 IF b = 0 THEN
2   RETURN a
3 ELSE
4   RETURN GCD(b, a mod b)
```

Με δεδομένους δύο φυσικούς αριθμούς α και β με α>β (αν ισχύει α<β αλλάζουμε τη σειρά τους):

- αν ο β είναι 0 τότε ο α είναι ο ΜΚΔ
- αν ο β δεν είναι 0 τότε επαναλαμβάνουμε τη διαδικασία[2] χρησιμοποιώντας τον β και το υπόλοιπο της διαίρεσης α/β

Παράδειγμα

Έστω ότι έχουμε τους αριθμούς 124, 34 και θέλουμε να βρούμε το μέγιστο κοινό διαιρέτη τους.

α	β	Επεξήγηση
124	34	124 > 34
34	22	22 = 124 mod 34 (δηλαδή το 22 είναι το υπόλοιπο του 124/34)
22	12	12 = 34 mod 22 (το 12 είναι το υπόλοιπο του 34/22)
12	10	κλπ.
10	2	
2	**0**	αφού το β γίνεται 0 ο αλγόριθμος τερματίζει και το α (εδώ το 2) είναι ο μέγιστος κοινός διαιρέτης

Επομένως ο αριθμός 2 είναι ο μέγιστος κοινός διαιρέτης (ΜΚΔ) των 124, 34.

Ο εκτεταμένος αλγόριθμος του Ευκλείδη ή Ευκλείδης2 είναι ο εξής:

Ο Μέγιστος Κοινός Διαιρέτης των ακεραίων a και b μπορεί να εκφραστεί ως [50]:

$ax + by = \gcd(a, b).$, όπου x, y δύο ακέραιοι

Ο ψευδοκώδικας του αλγορίθμου φαίνεται παρακάτω:

```
function extended_gcd(a, b)
    x := 0    lastx := 1
    y := 1    lasty := 0
    while b ≠ 0
        temp := b
```

```
        quotient := a div b
        b := a mod b
        a := temp

        temp := x
        x := lastx-quotient*x
        lastx := temp

        temp := y
        y := lasty-quotient*y
        lasty := temp
    return {lastx, lasty, a}
```

Παράδειγμα:

Υπολογισμός του ΜΚΔ των120 και 23.

Οι πράξεις που γίνονται φαίνονται παρακάτω:

Βήμα	**Πηλίκο**	**Υπόλοιπο**	**Αντικατάσταση**	**Νέοι όροι**
1		120		120 = 120 • 1 + 23 • 0
2		23		23 = 120 • 0 + 23 • 1
3	5	5 = 120 - 23 • 5	5 = (120 • 1 + 23• 0) - (120 • 0 + 23 • 1) • 5	5 = 120 • 1 + 23 • -5
4	4	3 = 23 - 5 • 4	3 = (120 • 0 + 23 • 1) - (120 • 1 + 23 • -5) • 4	3 = 120 • -4 + 23 • 21
5	1	2 = 5 - 3 • 1	2 = (120 • 1 + 23 • -5) - (120 • -4 + 23 • 21) • 1	2 = 120 • 5 + 23 • -26
6	1	1 = 3 - 2 • 1	1 = (120 • -4 + 23 • 21) - (120 • 5 + 23 • -26) • 1	1 = 120 • -9 + 23 • 47
7	2	0	**Τέλος**	

Έτσι στο τέλος έχουμε 1 = −9•120 + 47•23, άρα η λύση είναι: $x = -9$ και $y = 47$.

Επίσης, μπορούμε να πούμε ότι το −9 είναι το αντίστροφο του 120 modulo 23, και το 47 είναι το αντίστροφο του 23 modulo 120.

$-9 \cdot 120 \equiv 1 \textbf{ mod } 23$ και επίσης $47 \cdot 23 \equiv 1 \textbf{ mod } 120$.

Δ. Ψευδοκώδικας και παραδείγματα του SHA-1.

Ο ψευδοκώδικας του SHA-1 είναι ο εξής [52]:

```
Note 1: All variables are unsigned 32 bits and wrap modulo 2^32
when calculating
Note 2: All constants in this pseudo code are in big endian.
Within each word, the most significant bit is stored in the
leftmost bit position

Initialize variables:
h0 = 0x67452301
h1 = 0xEFCDAB89
h2 = 0x98BADCFE
h3 = 0x10325476
h4 = 0xC3D2E1F0

Pre-processing:
append the bit '1' to the message
append k bits '0', where k is the minimum number ≥ 0 such that
the resulting message
    length (in bits) is congruent to 448 (mod 512)
append length of message (before pre-processing), in bits, as
64-bit big-endian integer

Process the message in successive 512-bit chunks:
break message into 512-bit chunks
for each chunk
    break chunk into sixteen 32-bit big-endian words w[i], 0 ≤
i ≤ 15

    Extend the sixteen 32-bit words into eighty 32-bit words:
    for i from 16 to 79
        w[i] = (w[i-3] xor w[i-8] xor w[i-14] xor w[i-16])
leftrotate 1

    Initialize hash value for this chunk:
    a = h0
    b = h1
    c = h2
    d = h3
    e = h4

    Main loop:
```

```
    for i from 0 to 79
        if 0 ≤ i ≤ 19 then
            f = (b and c) or ((not b) and d)
            k = 0x5A827999
        else if 20 ≤ i ≤ 39
            f = b xor c xor d
            k = 0x6ED9EBA1
        else if 40 ≤ i ≤ 59
            f = (b and c) or (b and d) or (c and d)
            k = 0x8F1BBCDC
        else if 60 ≤ i ≤ 79
            f = b xor c xor d
            k = 0xCA62C1D6

        temp = (a leftrotate 5) + f + e + k + w[i]
        e = d
        d = c
        c = b leftrotate 30
        b = a
        a = temp

    Add this chunk's hash to result so far:
    h0 = h0 + a
    h1 = h1 + b
    h2 = h2 + c
    h3 = h3 + d
    h4 = h4 + e

Produce the final hash value (big-endian):
digest = hash = h0 append h1 append h2 append h3 append h4
```

Παραδείγματα:

SHA1("The quick brown fox jumps over the lazy dog")
= 2fd4e1c6 7a2d28fc ed849ee1 bb76e739 1b93eb12

Αλλάζοντας το d με το c στη λέξη dog έχουμε:

SHA1("The quick brown fox jumps over the lazy **c**og")
= de9f2c7f d25e1b3a fad3e85a 0bd17d9b 100db4b3

Ε. Κώδικας Υλοποίησης του ECDSA

Ο παρακάτω κώδικας περιέχει παραδείγματα στα σώματα της μορφής F_p (192-bit και 239-bit). Ο κώδικας που ακολουθεί είναι γραμμένος σε γλώσσα προγραμματισμού Java και το συγκεκριμένο αρχείο που τον περιέχει ονομάζεται

ECTest.java. Το αρχείο αυτό βρίσκεται στη βιβλιοθήκη και συγκεκριμένα στον υποφάκελο crypto-132\crypto-132\test\src\org\bouncycastle\crypto\test. Παρακάτω παρουσιάζεται ο κώδικας υλοποίησης του ECDSA στο σώμα F_p [54].

```
package org.bouncycastle.crypto.test;

import java.math.BigInteger;
import java.security.SecureRandom;

import org.bouncycastle.crypto.AsymmetricCipherKeyPair;
import org.bouncycastle.crypto.BasicAgreement;
import org.bouncycastle.crypto.agreement.ECDHBasicAgreement;
import org.bouncycastle.crypto.agreement.ECDHCBasicAgreement;
import org.bouncycastle.crypto.generators.ECKeyPairGenerator;
import org.bouncycastle.crypto.params.ECDomainParameters;
import
org.bouncycastle.crypto.params.ECKeyGenerationParameters;
import org.bouncycastle.crypto.params.ECPrivateKeyParameters;
import org.bouncycastle.crypto.params.ECPublicKeyParameters;
import org.bouncycastle.crypto.params.ParametersWithRandom;
import org.bouncycastle.crypto.signers.ECDSASigner;
import org.bouncycastle.math.ec.ECCurve;
import org.bouncycastle.math.ec.ECPoint;
import org.bouncycastle.util.encoders.Hex;
import org.bouncycastle.util.test.SimpleTest;

/**
 * ECDSA tests are taken from X9.62.
 */
public class ECTest
    extends SimpleTest
{
    /**
     * X9.62 - 1998,<br>
     * J.3.1, Page 152, ECDSA over the field Fp<br>
     * an example with 192 bit prime
     */
    private void testECDSA192bitPrime()
    {
        BigInteger r = new
BigInteger("3342403536405981729393488334694600415596881826869351677613");
        BigInteger s = new
BigInteger("5735822328888155254683894997897571951568553642892029982342");

        SecureRandom    k = new SecureRandom()
        {
            public void nextBytes(byte[] bytes)
            {
```

```
                byte[] vals = new
BigInteger("614050706706500106306506556566740556000616155656565
65656654").toByteArray();

                System.arraycopy(vals, vals.length-
bytes.length, bytes, 0, bytes.length);
            }
        };

        ECCurve.Fp curve = new ECCurve.Fp(
            new
BigInteger("627710173538668076383578942320766641608390870039032
4961279"), // q
            new
BigInteger("ffffffffffffffffffffffffffffffeffffffffffffffffc",
16), // a
            new
BigInteger("64210519e59c80e70fa7e9ab72243049feb8deecc146b9b1",
16)); // b

        ECDomainParameters params = new ECDomainParameters(
                curve,

curve.decodePoint(Hex.decode("03188da80eb03090f67cbf20eb43a188
00f4ff0afd82ff1012")), // G
                new
BigInteger("627710173538668076383578942317605901376719477318284
42284081")); // n

        ECPrivateKeyParameters priKey = new
ECPrivateKeyParameters(
            new
BigInteger("651056770906015076056810763456358567190100156695615
665659"), // d
            params);

        ParametersWithRandom param = new
ParametersWithRandom(priKey, k);

        ECDSASigner ecdsa = new ECDSASigner();

        ecdsa.init(true, param);

        byte[] message = new
BigInteger("968236873715988614170569073515315707566766479517")
.toByteArray();
        BigInteger[] sig = ecdsa.generateSignature(message);

        if (!r.equals(sig[0]))
        {
            fail("r component wrong." +
System.getProperty("line.separator")
```

```
                + " expecting: " + r +
System.getProperty("line.separator")
                + " got      : " + sig[0]);
        }

        if (!s.equals(sig[1]))
        {
            fail("s component wrong." +
System.getProperty("line.separator")
                + " expecting: " + s +
System.getProperty("line.separator")
                + " got      : " + sig[1]);
        }

        // Verify the signature
        ECPublicKeyParameters pubKey = new
ECPublicKeyParameters(

curve.decodePoint(Hex.decode("0262b12d60690cdcf330babab6e69763
b471f994dd702d16a5")), // Q
            params);

        ecdsa.init(false, pubKey);
        if (!ecdsa.verifySignature(message, sig[0], sig[1]))
        {
            fail("verification fails");
        }
    }

    private void decodeTest()
    {
        ECCurve.Fp curve = new ECCurve.Fp(
                new
BigInteger("627710173538668076383578942320766641608390870039032
4961279"), // q
                new
BigInteger("ffffffffffffffffffffffffffffffeffffffffffffffc",
16), // a
                new
BigInteger("64210519e59c80e70fa7e9ab72243049feb8deecc146b9b1",
16)); // b

        ECPoint p =
curve.decodePoint(Hex.decode("03188da80eb03090f67cbf20eb43a188
00f4ff0afd82ff1012"));

        if (!p.getX().toBigInteger().equals(new
BigInteger("188da80eb03090f67cbf20eb43a18800f4ff0afd82ff1012",
16)))
        {
            fail("x uncompressed incorrectly");
        }
```

```
        if (!p.getY().toBigInteger().equals(new
BigInteger("7192b95ffc8da78631011ed6b24cdd573f977a11e794811",
16)))
        {
            fail("y uncompressed incorrectly");
        }

        byte[] encoding = p.getEncoded();

        if (!areEqual(encoding,
Hex.decode("03188da80eb03090f67cbf20eb43a18800f4ff0afd82ff1012
")))
        {
            fail("point compressed incorrectly");
        }
    }

    /**
     * X9.62 - 1998,<br>
     * J.3.2, Page 155, ECDSA over the field Fp<br>
     * an example with 239 bit prime
     */
    private void testECDSA239bitPrime()
    {
        BigInteger r = new
BigInteger("308636143175167811492622547300668018854959378758531
778147462058306432176");
        BigInteger s = new
BigInteger("323813553209797357708078776831250505931891051755007
842781978505179448783");

        SecureRandom    k = new SecureRandom()
        {
            public void nextBytes(byte[] bytes)
            {
                byte[] vals = new
BigInteger("70000001756905664665550578175715710757050157577570
577957555657156756655").toByteArray();

                System.arraycopy(vals, vals.length-
bytes.length, bytes, 0, bytes.length);
            }
        };

        ECCurve.Fp curve = new ECCurve.Fp(
            new
BigInteger("883423532389192164791648750360308885314476597252960
362792450860609699839"), // q
            new
BigInteger("7fffffffffffffffffffff7fffffffffff8000000000007ffffffffffc", 16), // a
```

```
            new
BigInteger("6b016c3bdcf18941d0d654921475ca71a9db2fb27d1d377961
85c2942c0a", 16)); // b

        ECDomainParameters params = new ECDomainParameters(
            curve,

curve.decodePoint(Hex.decode("020ffa963cdca8816ccc33b8642bedf9
05c3d358573d3f27fbbd3b3cb9aaaf")), // G
            new
BigInteger("883423532389192164791648750360308884807550341691627
752275345424702807307")); // n

        ECPrivateKeyParameters priKey = new
ECPrivateKeyParameters(
            new
BigInteger("876300101507107567501066130761671078357010671067781
77671667167617872671 7"), // d
            params);

        ECDSASigner ecdsa = new ECDSASigner();
        ParametersWithRandom param = new
ParametersWithRandom(priKey, k);

        ecdsa.init(true, param);

        byte[] message = new
BigInteger("968236873715988614170569073515315707566766479517")
.toByteArray();
        BigInteger[] sig = ecdsa.generateSignature(message);

        if (!r.equals(sig[0]))
        {
            fail("r component wrong." +
System.getProperty("line.separator")
                + " expecting: " + r +
System.getProperty("line.separator")
                + " got      : " + sig[0]);
        }

        if (!s.equals(sig[1]))
        {
            fail("s component wrong." +
System.getProperty("line.separator")
                + " expecting: " + s +
System.getProperty("line.separator")
                + " got      : " + sig[1]);
        }

        // Verify the signature
        ECPublicKeyParameters pubKey = new
ECPublicKeyParameters(
```

```
curve.decodePoint(Hex.decode("025b6dc53bc61a2548ffb0f671472de6
c9521a9d2d2534e65abfcbd5fe0c70")), // Q
            params);

        ecdsa.init(false, pubKey);
        if (!ecdsa.verifySignature(message, sig[0], sig[1]))
        {
            fail("signature fails");
        }
    }



    /**
     * key generation test
     */
    private void testECDSAKeyGenTest()
    {
        SecureRandom    random = new SecureRandom();
        ECCurve.Fp curve = new ECCurve.Fp(
            new
BigInteger("883423532389192164791648750360308885314476597252960
362792450860609699839"), // q
            new
BigInteger("7fffffffffffffffffffff7fffffffffff8000000000007f
ffffffffc", 16), // a
            new
BigInteger("6b016c3bdcf18941d0d654921475ca71a9db2fb27d1d377961
85c2942c0a", 16)); // b

        ECDomainParameters params = new ECDomainParameters(
            curve,

curve.decodePoint(Hex.decode("020ffa963cdca8816ccc33b8642bedf9
05c3d358573d3f27fbbd3b3cb9aaaf")), // G
            new
BigInteger("883423532389192164791648750360308884807550341691627
752275345424702807307")); // n


        ECKeyPairGenerator          pGen = new
ECKeyPairGenerator();
        ECKeyGenerationParameters   genParam = new
ECKeyGenerationParameters(
                                        params,
                                        random);

        pGen.init(genParam);

        AsymmetricCipherKeyPair  pair =
pGen.generateKeyPair();
```

```
        ParametersWithRandom param = new
ParametersWithRandom(pair.getPrivate(), random);

        ECDSASigner ecdsa = new ECDSASigner();

        ecdsa.init(true, param);

        byte[] message = new
BigInteger("968236873715988614170569073515315707566766479517")
.toByteArray();
        BigInteger[] sig = ecdsa.generateSignature(message);

        ecdsa.init(false, pair.getPublic());

        if (!ecdsa.verifySignature(message, sig[0], sig[1]))
        {
            fail("signature fails");
        }
    }

    /**
     * Basic Key Agreement Test
     */
    private void testECBasicAgreementTest()
    {
        SecureRandom    random = new SecureRandom();
        ECCurve.Fp curve = new ECCurve.Fp(
            new
BigInteger("883423532389192164791648750360308885314476597252960362792450860609699839"), // q
            new
BigInteger("7fffffffffffffffffffff7fffffffffff8000000000007fffffffffc", 16), // a
            new
BigInteger("6b016c3bdcf18941d0d654921475ca71a9db2fb27d1d37796185c2942c0a", 16)); // b

        ECDomainParameters params = new ECDomainParameters(
            curve,

curve.decodePoint(Hex.decode("020ffa963cdca8816ccc33b8642bedf905c3d358573d3f27fbbd3b3cb9aaaf")), // G
            new
BigInteger("883423532389192164791648750360308884807550341691627752275345424702807307")); // n


        ECKeyPairGenerator          pGen = new
ECKeyPairGenerator();
        ECKeyGenerationParameters   genParam = new
ECKeyGenerationParameters(
                                        params,
```

```
                                        random);

        pGen.init(genParam);

        AsymmetricCipherKeyPair  p1 = pGen.generateKeyPair();
        AsymmetricCipherKeyPair  p2 = pGen.generateKeyPair();

        //
        // two way
        //
        BasicAgreement    e1 = new ECDHBasicAgreement();
        BasicAgreement    e2 = new ECDHBasicAgreement();

        e1.init(p1.getPrivate());
        e2.init(p2.getPrivate());

        BigInteger   k1 =
e1.calculateAgreement(p2.getPublic());
        BigInteger   k2 =
e2.calculateAgreement(p1.getPublic());

        if (!k1.equals(k2))
        {
            fail("calculated agreement test failed");
        }

        //
        // two way
        //
        e1 = new ECDHCBasicAgreement();
        e2 = new ECDHCBasicAgreement();

        e1.init(p1.getPrivate());
        e2.init(p2.getPrivate());

        k1 = e1.calculateAgreement(p2.getPublic());
        k2 = e2.calculateAgreement(p1.getPublic());

        if (!k1.equals(k2))
        {
            fail("calculated agreement test failed");
        }
    }



    public String getName()
    {
        return "EC";
    }

    public void performTest()
    {
```

```
        decodeTest();
        testECDSA192bitPrime();
        testECDSA239bitPrime();
        testECDSAKeyGenTest();
        testECBasicAgreementTest();
    }


    public static void main(
        String[]    args)
    {
        runTest(new ECTest());
    }
}
```

Βιβλιογραφία

1. Πουλάκης Δ. Μ., (2004), *Κρυπτογραφία: Η Επιστήμη της Ασφαλούς Επικοινωνίας*, Εκδόσεις Ζήτη.
2. Σταματίου Γ., (2004), *Κρυπτογραφία*, Ανάκτηση 8/8/2008, από World Wide Web: http://www.samos.aegean.gr/math/stamatiu/crypto_samos/
3. Κάτος Β. Α., Στεφανίδης Γ. Χ., (2003), *Τεχνικές κρυπτογραφίας & κρυπτανάλυσης*, Θεσσαλονίκη, Ζυγός.
4. Kurose J. F., Ross K. W., (2004), *Δικτύωση Υπολογιστών*, Αθήνα, Εκδόσεις Γκιούρδας.
5. Schneier, B., (1996), *Applied Cryptography*, 2nd Edition, New York, John Wiley & Sons.
6. IT Security ©, (2006), Ανάκτηση 12/8/2008 από World Wide Web: http://www.securitymanager.gr/it_security/protection_article.php.
7. IBM Website, (2001), Ανάκτηση 11/9/2008 από World Wide Web: http://www.ibm.com/developerworks/library/s-crypt02.html.
8. Certicom Inc., (2006), *Επίσημο site της Certicom*, Ανάκτηση 20/12/2008, από World Wide Web: http://www.certicom.com.
9. Ning P., Yin Y. L., (2001), *Efficient software implementation for finite field multiplication in normal basis*, ICICS, LNCS 2229.
10. Koblitz, N., (1987), *A Course in Number Theory and Cryptography*, Apringer-Verlag.
11. Blake I., Seroussi G., Smart N., Hitchin N., (1999), *Elliptic Curve in Cryptography*, London, Cambridge University Press.
12. Odlyzko A., (1999), *Discrete logarithms: The past and the future,* AT&T Labs – Research.
13. Odlyzko A., (1985), *Discrete logarithms in finite fields and their cryptographic significance*, New Jersey , AT&T Bell Laboratories Murray Hill.
14. Gaudry P., (2004), *Index calculus for abelian varieties and the elliptic curve discrete logarithm problem*, Εργαστήρια LIX, Πολυτεχνική Σχολή, France.

15. Κωνσταντίνου Ε., (2001), *Θεωρία και εφαρμογές ελλειπτικών καμπυλών στην ανάπτυξη κρυπτογραφικών συστημάτων*, Πανεπιστήμιο Πάτρας, Τμήμα Μηχανικών Η/Υ και Πληροφορικής.
16. Νεοφύτου Ο., (2004), *Κρυπτογραφικά συστήματα ελλειπτικών καμπυλών βασισμένα σε δυαδικά πεδία,* Πανεπιστήμιο Πάτρας, Τμήμα Μηχανικών Η/Υ και Πληροφορικής.
17. Certicom Research, (2000), SEC (Standards for Efficient Cryptography) 1: *Elliptic Curve Cryptography, Version 1.00*, Certicom Corporation.
18. Johnson D., Menezes A., (1999), *The elliptic curve digital signature algorithm (ECDSA),* University of Waterloo, Canada.
19. Hankerson D., Menezes A., Vanstone S., (2004), *Guide to Elliptic Curve Cryptography*, New York, Springer.
20. CS 431 lecture, (2000), *Computer Security*, Ανάκτηση 7/9/2008 από World Wide Web: http://134.193.15.25/vu/course/cs596A/
21. Kumar S., Girimondo M., Weimerskirch A., Paar C., Patel A., Wander A., (2001), *Embedded end-to-end wireless security with ECDH key exchange,* California, University of Michigan.
22. WIKIPEDIA: The free Encyclopedia, (2006), *ECDH*, Ανάκτηση 12/12/2008 από World Wide Web: http://en.wikipedia.org/wiki/ECDH.
23. WIKIPEDIA: The free Encyclopedia, (2006), *ECIES*, Ανάκτηση 12/12/2008 από World Wide Web: http://en.wikipedia.org/wiki/ECIES.
24. SUMMARY OF ANSI X9.63, *Public Key Cryptography for the Financial Services Industry: Key Agreement and Key Transport using Elliptic Curve Cryptography.*
25. Blake S., (2000), ANSI X9.63 Overview, *Key Agreement and Key Transport Using Elliptic Curve Cryptography*, Certicom.
26. Shoup V., (2001), *A proposal for an ISO standard for public key encryption - Version 2.1,* December 20.
27. Diffie W., Hellman M.E., (1976), New directions in cryptography. *IEEE Transactions on Information Theory*, IT-22(6):644-654.
28. WIKIPEDIA: The free Encyclopedia, (2006), *Public Key Cryptography*, Ανάκτηση 6/12/2008 από World Wide Web: http://en.wikipedia.org/wiki/Public-key_cryptography.

29. WIKIPEDIA: The free Encyclopedia, (2006), *Caesar cipher*, Ανάκτηση 8/12/2008 από World Wide Web: http://en.wikipedia.org/wiki/Caesar_cipher.
30. Tuchman W., (1997), *A brief history of the data encryption standard*, Internet besieged: countering cyberspace scofflaws: 275-280, ACM Press/Addison-Wesley Publishing Co. New York, NY, USA.
31. Feistel H., *Block Cipher Cryptographic System*, US Patent 3,798,359. Filed June 30, 1971. (IBM)
32. WIKIPEDIA: The free Encyclopedia, (2006), *Data Encryption Standard*, Ανάκτηση 12/12/2008 από World Wide Web: http://en.wikipedia.org/wiki/Data_Encryption_Standard.
33. Garon G., Outerbridge R., (1991), *DES Watch*, Cryptologia, Volume XV Number 3.
34. WIKIPEDIA: The free Encyclopedia, (2006), *International Data Encryption Algorithm*, Ανάκτηση 16/12/2008 από World Wide Web: http://en.wikipedia.org/wiki/International_Data_Encryption_Algorithm.
35. Garfinkel, Simson, (1994), *PGP: Pretty Good Privacy*. O'Reilly Media, pp.101–102, ISBN 978-1565920989.
36. Daemen J., Govaerts R., Vandewalle J., *Weak keys for IDEA*, CRYPTO '93, pp 224–231.
37. Biham E., Dunkelman O., Keller N., (2007), *A New Attack on 6-Round IDEA*, Springer-Verlag.
38. WIKIPEDIA: The free Encyclopedia, (2006), *International Data Encryption Algorithm*, Ανάκτηση 12/12/2008 από World Wide Web: http://en.wikipedia.org/wiki/Advanced_Encryption_Standard.
39. FIPS Publication 197, (2001), *Advanced Encryption Standard (AES)*, Federal Information Processing Standards.
40. NIST: National Institute of Standards and Technology, (2004), *Computer Security Objects Register (CSOR)*, Ανάκτηση 30/12/2008 από World Wide Web: http://csrc.nist.gov/groups/ST/toolkit/index.html.
41. Daemen J., Rijmen V., (1999), *AES Proposal: Rijndael, AES Algorithm Submission*, διαθέσιμο στο [40].

42. Γιαννακόπουλος Χ., (2007), *Γραμμική – Διαφορική Κρυπτανάλυση και κατασκευή Κρυπτογραφικά Ασφαλών S-boxes*, Πανεπιστήμιο Πάτρας, Τμήμα Μηχανικών Η/Υ και Πληροφορικής.
43. Discretix Technologies Ltd., (2007), *Introduction to Side Channel Attacks*, an industrial 'White Paper'.
44. Hathaway L., (2003), National Policy on the Use of the Advanced Encryption Standard (AES) to Protect National Security Systems and National Se, CNSS Policy No. 15, Fact Sheet No 1.
45. ALLADIN Security the Global Village, (2000), *The Enduring Value of Symmetric Encryption*, White Paper in www.eAladdin.com.
46. Biham E., Shamir A., (192), *Differential cryptanalysis of the full 16-round DES*, Advances in Cryptology, proceedings of CRYPTO '92, pp. 487-496.
47. Hellman M., (2002), *An Overview of Public Key Cryptography,* IEEE Communications Magazine.
48. WIKIPEDIA: The free Encyclopedia, (2006), Diffie-Hellman key exchange, Ανάκτηση 1/1/2009 από World Wide Web: http://en.wikipedia.org/wiki/Diffie-Hellman.
49. WIKIPEDIA: The free Encyclopedia, (2006), *RSA*, Ανάκτηση 10/10/2008 από World Wide Web: http://en.wikipedia.org/wiki/RSA.
50. WIKIPEDIA: The free Encyclopedia, (2006), *Extended Euclidean algorithm*, Ανάκτηση 2/1/2009 από World Wide Web: http://en.wikipedia.org/wiki/Extended_Euclidean_algorithm.
51. WIKIPEDIA: The free Encyclopedia, (2006), *Hash function*, Ανάκτηση 3/1/2009 από World Wide Web: http://en.wikipedia.org/wiki/Hash_function.
52. WIKIPEDIA: The free Encyclopedia, (2006), *SHA hash functions*, Ανάκτηση 3/1/2009 από World Wide Web: http://en.wikipedia.org/wiki/SHA.
53. Σαρηγιαννίδης Π., (2007), *Σημειώσεις για το μάθημα Ασφάλεια Υπολογιστικών και Επικοινωνιακών Συστημάτων* , Πανεπιστήμιο Ιωαννίνων Τμήμα Πληροφορικής.
54. The Legion Of The Bouncy Castle Copyright (c) 2000-2006, Ανάκτηση 14/4/2006 από World Wide Web: http://www.bouncycastle.org Copyright © 2000-2006.