Ανοικτό Πανεπιστήμιο Κύπρου

Σχολή Θετικών και Εφαρμοσμένων Επιστημών

Μεταπτυχιακή Διατριβή
στα Πληροφοριακά και Επικοινωνιακά Συστήματα

Δημιουργία Εικονικού Εργαστηρίου Κρυπτογραφίας και Ασφάλειας Επικοινωνιών

Δημήτρης Παπαδόπουλος

Επιβλέπων Καθηγητής
Κωνσταντίνος Λιμνιώτης

Σεπτέμβριος 2015

Ανοικτό Πανεπιστήμιο Κύπρου

Σχολή Θετικών και Εφαρμοσμένων Επιστημών

Δημιουργία Εικονικού Εργαστηρίου Κρυπτογραφίας και Ασφάλειας Επικοινωνιών

Δημήτρης Παπαδόπουλος

Επιβλέπων Καθηγητής
Κωνσταντίνος Λιμνιώτης

Η παρούσα μεταπτυχιακή διατριβή υποβλήθηκε
προς μερική εκπλήρωση των απαιτήσεων για απόκτηση

μεταπτυχιακού τίτλου σπουδών
στα Πληροφοριακά Συστήματα

από τη Σχολή Θετικών και Εφαρμοσμένων Επιστημών
του Ανοικτού Πανεπιστημίου Κύπρου

Σεπτέμβριος 2015

Περίληψη

Με τη ραγδαία πρόοδο των τεχνολογιών, το ζήτημα της ασφάλειας πληροφοριών και τηλεπικοινωνιών καθίσταται ολοένα και πιο κρίσιμο και επιτακτικό, ακριβώς γιατί εμφανίζονται ολοένα αυξανόμενοι κίνδυνοι επιθέσεων. Στον ακαδημαϊκό χώρο, στα Πανεπιστήμια του χώρου της Πληροφορικής δίνεται ξεχωριστή έμφαση στη διδασκαλία μαθημάτων που άπτονται της ασφάλειας τηλεπικοινωνιών: ακριβώς γιατί είναι ένα εξαιρετικά ευρύ και σύνθετο πεδίο, πολλές φορές προβλέπονται πλέον του ενός μαθημάτων στα προγράμματα σπουδών, κάθε ένα εκ των οποίων καλύπτει ξεχωριστή οπτική (π.χ. ύπαρξη αυτόνομου μαθήματος ειδικά για την κρυπτογραφία). Πέρα όμως από το πλούσιο θεωρητικό υπόβαθρο που καλύπτει τα εν λόγω ζητήματα, η πρακτική εφαρμογή των εν λόγω θεμάτων από την πλευρά των φοιτητών αποκτά ιδιαίτερη σημασία για την πλήρη κατανόηση και εμβάθυνση στις σχετικές έννοιες. Ωστόσο, η πρακτική εφαρμογή εμφανίζει κατά κανόνα δυσκολίες, ακριβώς γιατί πρέπει να δομηθεί ένα κατάλληλο περιβάλλον προσομοίωσης, έτσι ώστε οι επιθέσεις ασφαλείας που θα πραγματοποιούνται για εκπαιδευτικούς σκοπούς να έχουν όλα τα χαρακτηριστικά των αυθεντικών, χωρίς όμως να βρίσκονται στο πραγματικό περιβάλλον (π.χ. σε πραγματικές ιστοσελίδες).

Αντικείμενο της παρούσας διατριβής είναι η ανάπτυξη ενός εικονικού εργαστηρίου κρυπτογραφίας και ασφάλειας πληροφοριών, το οποίο θα μπορεί να αξιοποιηθεί στη διδασκαλία σχετικών μαθημάτων σε προπτυχιακούς και μεταπτυχιακούς φοιτητές – κατάλληλο δε και σε περιπτώσεις εξ αποστάσεων εκπαίδευσης, όπως είναι η περίπτωση του Ανοικτού Πανεπιστημίου Κύπρου. Ειδικότερα, αναπτύχθηκε μια πλατφόρμα εφαρμογών, εύκολα επεκτάσιμη, οι οποίες ομαδοποιήθηκαν και συνδέθηκαν σε μια κεντρική ιστοσελίδα, μέσω των οποίων ο φοιτητής μπορεί να πειραματισθεί και εξασκηθεί σε επιθέσεις και απειλές κατά συστημάτων. Στο εν λόγω λογισμικό, το οποίο εντάσσεται στην κατηγορία εκπαιδευτικού λογισμικού προσομοίωσης, καλύπτονται κάποια είδη κύριων επιθέσεων ασφαλείας, όπως η υποκλοπή δεδομένων λόγω μη ισχυρού κρυπτογραφικού αλγορίθμου (χρήση ακολουθίας κλειδιού με όχι καλά χαρακτηριστικά τυχαιότητας), ανίχνευση συνθηματικών, καθώς και διαδικτυακές επιθέσεις τύπου SQL “injection” σε συστήματα δεδομένων. Στην παρούσα διατριβή, πέραν της αναλυτικής παρουσίασης του εικονικού εργαστηρίου, των τεχνολογιών που χρησιμοποιήθηκαν και του τρόπου χρήσης του, επεξηγούνται επίσης όλες οι σχετικές έννοιες, καθώς και οι τρόποι αντιμετώπισης των εν λόγω επιθέσεων.

Summary

The increasingly rapid evolution and growth in the complexity of new systems and technologies, coupled with the appearance of powerful threats, present demanding challenges for maintaining the security of Information and Communications Technology (ICT) systems. Hence, due to the significance of ICT security, the syllabus of almost any academic Computer Science department includes at least one course lying in the general field of data security. Apart though from the rich mathematical and theoretical background that needs to be covered, mounting attacks in practice is crucial for a deep understanding of their principles and, as a result, to conclude with the necessary countermeasures to thwart them. From the educational perspective, the challenging task is to develop a simulation environment that fully resembles the original systems, in order to perform security attacks.

In this thesis, a virtual lab on data security and cryptography has been developed, aiming to serve as a basic educational tool for either undergraduate or postgraduate students in a ICT security academic course. It should be pointed out that this virtual lab can be utilized for courses that are based on distance learning, as those offered by the Open University of Cyprus. More precisely, an easily extensible software suite has been developed, which can be used by students to mount specific types of attacks on systems. This tool, which is actually a simulation educational tool, covers some of the most important types of attacks, namely data eavesdropping due to weak encryption that occurs from using a keystream with pure cryptographic properties, password cracking and SQL injection attacks. This thesis describes all the used technologies for developing this virtual lab, provides guidelines and examples for using the lab, whereas it also studies all the aforementioned types of attacks and the relative countermeasures.

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον καθηγητή μου, Λιμνιώτη Κωνσταντίνο για την καθοδήγηση και τις συμβουλές που μου έδινε κατά την διάρκεια της εκπόνησης της διατριβής αυτής.

Επίσης, θα ήθελα να αφιερώσω τη διατριβή αυτή στην οικογένεια μου. Ευχαριστώ για την υπομονή που δείξατε.

Περιεχόμενα

Κεφάλαιο 1
Εισαγωγή

Σε μια εποχή στην οποία το Διαδίκτυο είναι η κατ' εξοχήν πηγή πληροφοριών για τον άνθρωπο σε όλες τις πτυχές της ζωής του (εκπαιδευτικά, επαγγελματικά, προσωπικά), αλλά και απαραίτητο εργαλείο για καθημερινές συναλλαγές, είναι επόμενο να εμφανιστούν φαινόμενα επιθέσεων και παραβιάσεων συστημάτων. Η ραγδαία εξέλιξη της τεχνολογίας, με την όλο και πιο άμεση πρόσβαση στο Διαδίκτυο από συσκευές οι οποίες το επιτρέπουν, αψηφώντας την τοποθεσία και τις μετακινήσεις, η χρήση τους από χρήστες όλων των ηλικιών και μορφωτικών επιπέδων, θέτει σε κίνδυνο την εμπιστευτικότητα και ακεραιότητα των δεδομένων και επιτακτική ανάγκη την ασφάλειά τους από τυχόν επιθέσεις. Υπάρχουν πολλά παραδείγματα τα τελευταία χρόνια από επιθέσεις μεγάλων διαστάσεων που έθεσαν σε κίνδυνο δεδομένα μεγάλων εταιρειών και οργανισμών.

Ως άμεση απόρροια των ανωτέρω, η ακαδημαϊκή κοινότητα δίνει μεγάλη έμφαση σε θέματα ασφάλειας δεδομένων. Στη συντριπτική πλειοψηφία των πανεπιστημιακών προγραμμάτων σπουδών Πληροφορικής συμπεριλαμβάνονται μαθήματα σχετικά με την ασφάλεια δεδομένων αλλά και ιδιαίτερα του Διαδικτύου. Πέρα όμως από το πλούσιο θεωρητικό υπόβαθρο που διέπει τις έννοιες που άπτονται της ασφάλειας των δεδομένων, είναι απαραίτητη η πρακτική εξάσκηση και εφαρμογή για το φοιτητή, προκειμένου να εντρυφήσει στα εν λόγω ζητήματα.

Αντικείμενο της παρούσας διατριβής είναι η δημιουργία ενός εικονικού εργαστηρίου κρυπτογραφίας και ασφάλειας πληροφοριών, στο οποίο ο φοιτητής θα αξιοποιεί ένα περιβάλλον προσομοίωσης επιθέσεων για την επί τόπου μελέτη και αντιμετώπισή τους. Με αυτόν τον τρόπο, θα είναι διαθέσιμο προς τους φοιτητές ένα εκπαιδευτικό λογισμικό, κατάλληλα προσανατολισμένο σε ζητήματα ασφάλειας, γεγονός το οποίο θα συνεισφέρει ουσιαστικά στην εκπαιδευτική διαδικασία. Το εν λόγω λογισμικό προσφέρεται και για εξ αποστάσεως εκπαίδευση, ως εκ τούτου θα μπορούσε να αξιοποιηθεί και στις σχετικές Θεματικές Ενότητες του Ανοικτού Πανεπιστημίου Κύπρου. Για την ανάπτυξή του λήφθηκαν υπόψη σημαντικές επιθέσεις ασφαλείας που πρέπει να διδαχθούν στους φοιτητές και να παρουσιαστούν στην πράξη, ενώ επίσης λήφθηκε μέριμνα ώστε να είναι επεκτάσιμο και τροποποιήσιμο από τον διδάσκοντα.

Ειδικότερα, η δομή της παρούσας διατριβής είναι η εξής:

Στο **κεφάλαιο 2** πραγματοποιείται μία επισκόπηση της σημερινής κατάστασης στην ασφάλεια δικτύων και πληροφοριακών συστημάτων, καταδεικνύοντας τη σπουδαιότητά της.

Στα κεφάλαια 3 – 5 περιγράφονται βασικές επιθέσεις ασφαλείας, οι οποίες αποτελούν και τον "κορμό" του λογισμικού που αναπτύχθηκε στο πλαίσιο της διατριβής. Ειδικότερα, στο **κεφάλαιο 3** περιγράφονται βασικές έννοιες της κρυπτογραφίας με έμφαση στους αλγόριθμους ροής, στη γραμμική πολυπλοκότητα της κλειδοροής (που είναι στενά συνυφασμένη με την προβλεψιμότητα αυτής) και στις επιθέσεις γνωστού κειμένου. Το **κεφάλαιο 4** ασχολείται με την ασφάλεια των συνθηματικών, τις μεθόδους που χρησιμοποιούνται για ανίχνευση συνθηματικών και με τους τρόπους αντιμετώπισής τους. Στο **κεφάλαιο 5** περιγράφονται οι επιθέσεις "έγχυσης" κώδικα SQL, (SQL injection) και εισαγωγής κώδικα μέσω ιστοσελίδων (cross site scripting ή XSS), οι οποίες αποτελούν τον μεγαλύτερο κίνδυνο σήμερα για τις διαδικτυακές εφαρμογές.

Το **κεφάλαιο 6** αποτελεί το κύριο κεφάλαιο της διατριβής και περιγράφει αναλυτικά το εικονικό εργαστήριο το οποίο δημιουργήθηκε στα πλαίσια εκπόνησης αυτής. Περιγράφονται τα εργαλεία λογισμικού που χρησιμοποιήθηκαν, ο τρόπος χρήσης τους, ενώ δίνονται και αναλυτικά παραδείγματα.

Τέλος, σύνοψη και συμπεράσματα της διατριβής αποτυπώνονται στο **κεφάλαιο 7**.

Κεφάλαιο 2
Ασφάλεια Δικτύων και Πληροφοριακών Συστημάτων

Η σημερινή μοντέρνα κοινωνία και οι θεσμοί της δεν μπορούν λειτουργήσουν χωρίς επικοινωνίες πληροφοριακών συστημάτων. Η μεγάλη και απότομη ανάπτυξη των πληροφοριακών συστημάτων, αλλά και των τηλεπικοινωνιακών δικτύων, κατέστησε οργανισμούς και πρόσωπα να εξαρτώνται σχεδόν απόλυτα, πλέον, από τις πληροφορίες που διακινούνται σε αυτά [21], [30]. Πολλοί, αν όχι όλοι, οι οργανισμοί βασίζονται όλο και περισσότερο σε δίκτυα και πληροφοριακά συστήματα για να αποθηκεύσουν και διακινήσουν πολύτιμα εταιρικά δεδομένα και ευαίσθητες πληροφορίες πελατών τους. Επιπρόσθετα, όλοι εμείς, ατομικά, στηριζόμαστε σε τεχνολογίες πληροφοριακών συστημάτων και διαδικτύου για να αποθηκεύουμε ευαίσθητα και πολύτιμα προσωπικά δεδομένα και πληροφορίες που αφορούν την καθημερινότητά μας. Αυτά μπορεί να περιλαμβάνουν φακέλους δεδομένων σε υπηρεσίες υπολογιστικού νέφους (cloud computing), συνδρομές σε διαδικτυακές εφαρμογές όπου αποθηκεύουμε φωτογραφίες δικές μας και της οικογένειάς μας, προσωπική ηλεκτρονική αλληλογραφία και πολλά άλλα.

Τα πληροφορικά συστήματα αυτά διακινούν τα δεδομένα μέσα από το Διαδίκτυο και όλες οι πληροφορίες είναι διαθέσιμες – κυριολεκτικά – από οποιοδήποτε σημείο του πλανήτη,

οποιοσδήποτε και αν τις χρειαστεί, ή αυτός είναι κάποιο εξουσιοδοτημένο εταιρικό στέλεχος, ή είναι κάποιο μέλος της οικογένειας που κατοικεί σε μια άλλη χώρα.

Όσο ο όγκος και η πολυτιμότητα των δεδομένων αυτών αυξάνει τόσο αυξάνονται και οι επιθέσεις στα συστήματα αυτά. Αυξάνεται, συνεπώς, και η ανάγκη για την προστασία των συστημάτων. Τα τελευταία χρόνια, μεγάλοι οργανισμοί έχουν πέσει θύματα επιθέσεων οι οποίες επέφεραν τεράστια οικονομική ζημιά στους οργανισμούς αυτούς, αλλά και επηρέασαν εκατομμύρια ανθρώπους των οποίων τα δεδομένα έχουν κλαπεί.

2.1 Γνωστές επιθέσεις ασφαλείας

Ένα από τα σημαντικότερα παραδείγματα αποτελεί η επίθεση στα καταστήματα ρούχων TJX Companies Inc. το 2005 και 2006 [19]. Σε αυτή την επίθεση οι επιτιθέμενοι απέκτησαν πρόσβαση στο δίκτυο της εταιρίας μέσω των ασύρματων δικτύων σε τοπικά υποκαταστήματα, από όπου και κατάφερναν να παρακολουθούν την κίνηση των πακέτων δεδομένων που κινούνταν στο εταιρικό δίκτυο. Αν και τα δεδομένα διακινούνταν στο δίκτυο κρυπτογραφημένα, **η κρυπτογράφηση τους δεν ήταν αρκετά ισχυρή** και συνεπώς ανεπαρκής και ευάλωτη. Οι επιτιθέμενοι κατάφεραν να κλέψουν 45 εκατομμύρια αριθμούς πιστωτικών καρτών και ευαίσθητα προσωπικά δεδομένα μισού, περίπου, εκατομμυρίου πελατών των καταστημάτων (οι οποίοι απλά έκαναν κανονικά τις αγορές τους). Η εταιρία αναγκάστηκε να πληρώσει 41 εκατομμύρια δολάρια στις τράπεζες των κλαπέντων πιστωτικών καρτών, 880,000 δολάρια σε πρόστιμο προς την εταιρία Visa ενώ υπήρξε πτώση 256 εκατομμυρίων δολαρίων πτώση στα κέρδη του επόμενου χρόνου. Το τελικό κόστος (περιλαμβανομένων προστίμων προς τις αρχές και αποζημιώσεων προς πελάτες) εκτιμάται να φτάνει κοντά στο 1 δις δολάρια.

Οι επιθέσεις αυτού του μεγέθους συνεχίζονται μέχρι και σήμερα, και μεγάλες εταιρίες ακόμη δεν κατάφεραν να θωρακιστούν επαρκώς. Το ινστιτούτο Ponemon αναφέρει χαρακτηριστικά σε έκθεση του ότι "*…το 2014 ήταν η χρονιά των χειρότερων επιθέσεων και παραβιάσεων*" [24]. Η έκθεση αυτή εστιάζει σε επιθέσεις που έγιναν τον χρόνο που πέρασε, κατατάσσοντας τις με κριτήριο τον αριθμό των ατόμων που επηρεάστηκαν από αυτές. Οι πέντε πιο σημαντικές επιθέσεις του 2014, σύμφωνα πάντα με την έκθεση Ponemon είναι:

- Καταστήματα Target (40 εκατομμύρια αριθμοί πιστωτικών καρτών και 70 εκατομμύρια εγγραφές προσωπικών δεδομένων πελατών των καταστημάτων).

- eBay (επηρεάστηκαν 145 εκατομμύρια άτομα).

- JPMorgan Chase & Co. (Επηρεάστηκαν 76 εκατομμύρια νοικοκυριά και 7 εκατομμύρια μικρές επιχειρήσεις).

- Home Depot (56 εκατομμύρια αριθμοί πιστωτικών καρτών).

- CHS community Health Systems (επηρεάστηκαν 4.5 εκατομμύρια άνθρωποι).

Δύο άλλοι κορυφαίοι οργανισμοί που παρακολουθούν τα θέματα ασφάλειας δικτύων και πληροφοριακών συστημάτων συντάσσουν τους δικούς τους καταλόγους απειλών διαδικτύου και επικινδυνότητας επιθέσεων. Σε αυτούς τους καταλόγους μπορούμε να δούμε την επικινδυνότητα από τη διάσταση του είδους των επιθέσεων. Οι κατάλογοι αυτοί είναι ο "Top 25 Most Dangerous Software Errors" των MITRE/SANS [05] και ο " Top 10 List " των OWASP [17].

Σύμφωνα με αυτούς τους οργανισμούς, οι κορυφαία απειλή που αντιμετωπίζει σήμερα ο τομέας των δικτύων και πληροφοριακών συστημάτων είναι η επίθεση τύπου "έγχυσης SQL" (SQL Injection). Η επίθεση αυτή τοποθετείται στην πρώτη θέση των καταλόγων αυτών και αποτελεί τον πιο δημοφιλή τρόπο με τον οποίο ο επιτιθέμενος θα προσπαθήσει να αποκτήσει κάποιο συνθηματικό το οποίο θα του δώσει πρόσβαση στην εφαρμογή ή στο υποκείμενο σύστημα.

Η μελέτη των επιθέσεων που γίνονται τα τελευταία χρόνια από τους τρεις αυτούς οργανισμούς, καταδεικνύει ότι για να αποκτήσει κάποιος κακόβουλος επιτιθέμενος πρόσβαση σε κάποιο σύστημα ή δίκτυο, προσπαθεί, συνήθως, να το επιτύχει μέσω κάποιου συνθηματικού – αυτό μπορεί να εξηγηθεί αν αναλογιστεί κανείς ότι η χρήση συνθηματικών παραμένει ο πιο δημοφιλής μηχανισμός αυθεντικοποίησης των χρηστών. Μία ισχυρή και ευρέως διαδεδομένη μέθοδος που χρησιμοποιείται είναι η επίθεση SQL Injection, ενώ όταν τα συνθηματικά δεν γίνουν γνωστά με αυτό τον τρόπο, εκτελούνται άλλες επιθέσεις, οι οποίες βασίζονται σε μεθόδους ανίχνευσης συνθηματικών (διάφοροι τύποι επιθέσεων λεξικού). Παρατηρείται επίσης ότι, αφού ο επιτιθέμενος αποκτήσει πρόσβαση σε κάποιο σύστημα ή δίκτυο, η προσπάθεια του επικεντρώνεται στην υποκλοπή δεδομένων τα οποία είτε διακινούνται στο δίκτυο είτε βρίσκονται αποθηκευμένα κάπου. Στις περιπτώσεις όπου τα δεδομένα αυτά δεν κρυπτογραφούνται καθόλου ή κρυπτογραφούνται με τη χρήση μη ισχυρού κρυπτογραφικού αλγορίθμου, ο επιτιθέμενος επιτυγχάνει το σκοπό του, και οι οργανισμοί ασφάλειας καταγράφουν ακόμη ένα περιστατικό υποκλοπής.

Διαπιστώνεται λοιπόν ότι η ανάγκη προστασίας και ασφάλειας των δικτύων και πληροφοριακών συστημάτων είναι πολύ μεγάλη. Οι επιθέσεις των τελευταίων χρόνων αποδεικνύουν ότι υπάρχει πολύς δρόμος να καλυφτεί στον τομέα αυτό, γενικά, αλλά και πιο συγκεκριμένα είναι ανάγκη να επικεντρωθούμε στην άμυνα ενάντια σε δημοφιλής επιθέσεις και ευπάθειες όπως είναι η επίθεση SQL injection, οι μέθοδοι ανίχνευσης συνθηματικών (και η εκπαίδευση προς τη σωστή τους χρήση), αλλά και η υποκλοπή δεδομένων λόγω μη ισχυρού κρυπτογραφικού αλγορίθμου.

2.2 Νομικές υποχρεώσεις υπευθύνων

Κλείνοντας, προσθέτουμε ότι η ανάγκη για προστασία των συστημάτων και δικτύων γίνεται ακόμη πιο επιτακτική μέσω των νομικών διατάξεων των αρχών των διαφόρων κρατών, οι οποίες θέτουν πλέον τις υποχρεώσεις των οργανισμών σε νομικό επίπεδο. Οι υπεύθυνοι, πέραν του γεγονότος ότι ένα συμβάν ασφαλείας θα πλήξει την αξιοπιστία και τη φήμη τους, υποχρεώνονται και νομικά στο να λαμβάνουν τα κατάλληλα μέτρα για την ασφάλεια της επεξεργασίας. Εξάλλου, όπως αναφέρθηκε και πιο πάνω, οργανισμοί οι οποίοι δέχθηκαν επιθέσεις, αναγκάζονται να πληρώσουν τεράστια ποσά σε πρόστιμα προς τις αρχές για παραβιάσεις των διατάξεων αυτών.

Στο σημείο αυτό θα σταθούμε για λίγο στο νομικό πλαίσιο προστασίας προσωπικών δεδομένων που διέπει τα Κράτη – Μέλη της Ευρωπαϊκής Ένωσης. Ειδικότερα, τα Κράτη – Μέλη έχουν ενσωματώσει, στην εθνική τους νομοθεσία, την Ευρωπαϊκή Οδηγία 95/46/ΕΚ σχετικά με την επεξεργασία των προσωπικών δεδομένων [08]. Ήδη στην Οδηγία αυτή τίθεται ένα γενικό πλαίσιο υποχρεώσεων των υπευθύνων επεξεργασίας, αλλά και των συνεργατών/αναδόχων αυτών (εκτελούντες την επεξεργασία, όπως ονομάζονται). Για παράδειγμα, στην Οδηγία αυτή επισημαίνεται ότι ο υπεύθυνος επεξεργασίας οφείλει να λαμβάνει τα κατάλληλα οργανωτικά, τεχνικά και φυσικά μέτρα για την ασφάλεια των δεδομένων και την προστασία τους από τυχαία ή αθέμιτη καταστροφή, τυχαία απώλεια, αλλοίωση, απαγορευμένη διάδοση ή πρόσβαση και κάθε άλλη μορφή αθέμιτης επεξεργασίας, καθώς επίσης και ότι τα μέτρα αυτά πρέπει να εξασφαλίζουν επίπεδο ασφαλείας ανάλογο προς τους κινδύνους που συνεπάγεται η επεξεργασία και η φύση των δεδομένων.

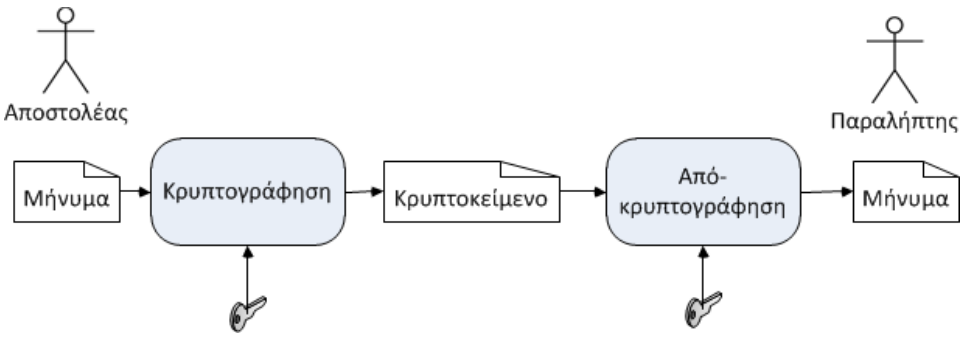
Ωστόσο, η ανωτέρω Οδηγία 95/46/ΕΚ επίκειται να αντικατασταθεί από έναν νέο Κανονισμό για την προστασία των προσωπικών δεδομένων. Ο Κανονισμός αυτός θα έχει άμεση ισχύ στα Κράτη

– Μέλη (δηλαδή θα αντικαταστήσει τις υπάρχουσες νομοθεσίες) και είναι ακόμα πιο αυστηρός στην προστασία των προσωπικών δεδομένων. Μεταξύ των νέων στοιχείων που επιφέρει ο νέος Κανονισμός, με βάση τη νυν μορφή του σχεδίου αυτού [07], είναι ότι όλοι οι φορείς (εταιρείες, οργανισμοί κτλ.) θα υποχρεούνται να γνωστοποιούν στις αρμόδιες αρχές τις σοβαρές παραβιάσεις δεδομένων που λαμβάνουν χώρα χωρίς καθυστέρηση, εντός 24 ωρών. Με αυτόν τον τρόπο οι χρήστες προστατεύονται περισσότερο, γιατί θα λαμβάνονται άμεσα μέτρα για την αντιμετώπιση των δυσμενών συνεπειών που θα επιφέρει μια διαρροή των δεδομένων τους.

Ως εκ τούτου, γίνεται σαφές ότι όλοι οι οργανισμοί είναι υποχρεωμένοι να επενδύουν συνεχώς σε θέματα ασφάλειας, σε μια προσπάθεια να διασφαλίσουν την μέγιστη δυνατή προστασία των δεδομένων που επεξεργάζονται.

Κεφάλαιο 3
Κρυπτογραφία

Η κρυπτογραφία είναι ένα σύνολο τεχνικών οι οποίες έχουν στόχο τη διασφάλιση των θεμάτων ασφάλειας πληροφοριών, όπως είναι η εμπιστευτικότητα, η ακεραιότητα των δεδομένων, η πιστοποίηση αυθεντικότητας κ.τ.λ. [12]. Αφορά, κυρίως τη μεταφορά μηνυμάτων μέσω κάποιου μη ασφαλούς καναλιού μετάδοσης, με τη διασφάλιση ότι τα μηνύματα αυτά θα φτάσουν στον προορισμό τους ικανοποιώντας τις πιο πάνω αρχές ασφαλείας.



Σχήμα 3.1 – Γενικότερο πρότυπο κρυπτογραφικής τεχνικής

Υπάρχουν πολλές τεχνικές και αλγόριθμοι κρυπτογράφησης που επιχειρούν να επιτύχουν το στόχο αυτό, και ακολουθούν μία γενικότερη λογική. Συγκεκριμένα, το μήνυμα κρυπτογραφείται

(δηλαδή, μετασχηματίζεται σε μία ακατάληπτη μορφή) πριν την αποστολή του, ενώ κατά την παραλαβή του από το νόμιμο αποδέκτη του αποκρυπτογραφείται (ανακτάται στην αρχική του μορφή). Η κρυπτογράφηση και αποκρυπτογράφηση των μηνυμάτων απαιτούν τη χρήση κλειδιού ή κλειδιών, ανάλογα με την τεχνική. Επίσης, για την ανάπτυξη και αποτίμηση κρυπτογραφικών αλγορίθμων ως προς την ασφάλεια που παρέχουν, κάνουμε πάντα τις εξής υποθέσεις:

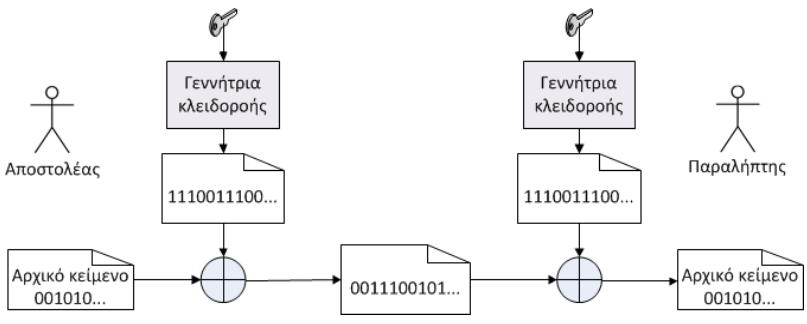
- Το κανάλι μετάδοσης είναι πάντα ανοικτό, άρα κάποιος υποκλοπέας έχει ελεύθερη πρόσβαση στο κρυπτοκείμενο.

- Ο αλγόριθμος κρυπτογράφησης και αποκρυπτογράφησης που χρησιμοποιείται είναι γνωστός σε όλους, ακόμη και σε κάποιο υποκλοπέα. Μόνο το αρχικό μήνυμα και το κλειδί είναι μυστικά (**Αρχή του Kerchoff** [42]).

Οι κρυπτογραφικοί αλγόριθμοι που χρησιμοποιούνται σήμερα χωρίζονται σε αλγόριθμους συμμετρικού κλειδιού και αλγόριθμους δημόσιου (ή ασύμμετρου) κλειδιού. Στους αλγόριθμους συμμετρικού κλειδιού χρησιμοποιείται το ίδιο κλειδί κατά την κρυπτογράφηση και αποκρυπτογράφηση, ενώ στους αλγόριθμους δημόσιου κλειδιού χρησιμοποιούνται δύο διαφορετικά κλειδιά, το ένα κατά την κρυπτογράφηση και το άλλο κατά την αποκρυπτογράφηση.

3.1 Αλγόριθμοι ροής

Οι αλγόριθμοι ροής (stream ciphers) υπάγονται στην κατηγορία των αλγορίθμων συμμετρικού κλειδιού. Χρειάζεται, δηλαδή, να χρησιμοποιηθεί κάποιο κοινό κλειδί τόσο κατά την κρυπτογράφηση όσο και κατά την αποκρυπτογράφηση.



Σχήμα 3.2 – Αλγόριθμος ροής

Με αυτό τον αλγόριθμο, κρυπτογραφούνται τα ψηφία (bits) του αρχικού μηνύματος ένα προς ένα, ξεχωριστά, χρησιμοποιώντας μια ροή ψευδοτυχαίων ψηφίων (κλειδοροή). Τα bits αυτής της ψευδοτυχαίας ακολουθίας προστίθενται (XOR) με τα bits του αρχικού μηνύματος, παράγοντας έτσι το κρυπτοκείμενο. Κατά την αποκρυπτογράφηση χρησιμοποιείται η ίδια κλειδοροή, με την οποία προσθέτουμε ένα-ένα ξεχωριστά τα ψηφία του κρυπτοκειμένου αυτή τη φορά. Το αποτέλεσμα είναι η αποκρυπτογράφηση του κρυπτοκειμένου στα ψηφία του αρχικού κειμένου.

Για παράδειγμα, με κλειδοροή 01110010 και αρχικό κείμενο 01011101, κρυπτογραφούμε:

```
    01011101 αρχικό κείμενο
XOR 01110010 κλειδοροή
    --------
    00101111 κρυπτοκείμενο
```

και αποκρυπτογραφούμε:

```
    00101111 κρυπτοκείμενο
XOR 01110010 κλειδοροή
    --------
    01011101 αρχικό κείμενο
```

Τα ψηφία της κλειδοροής παράγονται από τη γεννήτρια κλειδοροής την οποία έχουν στην διάθεσή τους αποστολέας και παραλήπτης. Η γεννήτρια κλειδοροής χρησιμοποιείται κατά την κρυπτογράφηση και αποκρυπτογράφηση και, ανάλογα με τη γεννήτρια, παράγει ακολουθίες ψηφίων οι οποίες έχουν συγκεκριμένη περίοδο (αφού η γεννήτρια είναι πεπερασμένου μεγέθους, θα υπάρχει αναπόφευκτα επαναληπτικότητα της κλειδοροής). Η αρχική κατάσταση της γεννήτριας αποτελεί το μυστικό κλειδί που ανταλλάσσουν τα δύο συνδιαλεγόμενα μέρη. Στην περίπτωση που το μήνυμα το οποίο κρυπτογραφείται είναι μεγαλύτερο από την περίοδο της ακολουθίας τότε η ακολουθία επαναλαμβάνεται περιοδικά μέχρι να ικανοποιηθεί το μέγεθος του κειμένου.

Οι κρυπτογραφικοί αλγόριθμοι ροής, λόγω της απλότητάς τους, προτιμώνται σε εφαρμογές όπου υπάρχουν απαιτήσεις για υψηλή ταχύτητα κρυπτογράφησης και χαμηλή κατανάλωση ισχύος (π.χ. ασύρματα δίκτυα αισθητήρων, κινητές επικοινωνίες). Πρέπει πάντως να σημειωθεί ότι ακόμα στο Διαδίκτυο, για πολλά χρόνια τα πρωτόκολλα ασφαλείας SSL/TLS παρείχαν τη

δυνατότητα υλοποίησης κρυπτογραφικού αλγορίθμου ροής για την ασφαλή μετάδοση των δεδομένων – συγκεκριμένα, τον αλγόριθμο RC4 [26]. Ακόμα και σήμερα, υπάρχουν διαδικτυακοί τόποι που το πρωτόκολλο ασφαλείας τους υλοποιεί τον RC4.

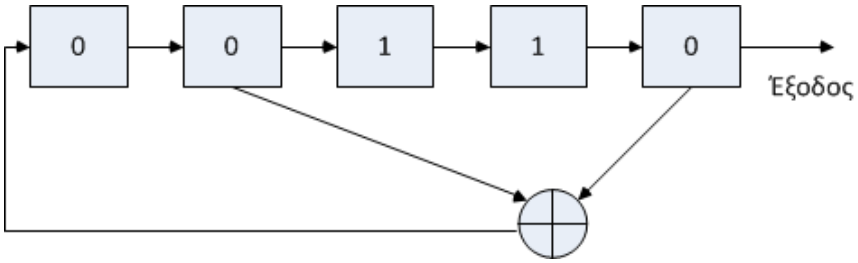
3.1.1 Ασφάλεια αλγορίθμων ροής

Αν κάποιος υποκλοπέας του κρυπτομηνύματος καταφέρει να προβλέψει την κλειδοροή με την οποία κρυπτογραφήθηκε το μήνυμα, θα μπορέσει να το αποκρυπτογραφήσει ως να ήταν ο νόμιμος παραλήπτης του. Η ασφάλεια λοιπόν των αλγορίθμων ροής εξαρτάται από το πόσο προβλέψιμη είναι η κλειδοροή η οποία χρησιμοποιείται κατά την κρυπτογράφηση. Ακολουθίες με "αδύναμα" χαρακτηριστικά ασφάλειας, μπορούν να προβλεφτούν με τη χρήση επιθέσεων στο κρυπτοκείμενο (το οποίο μεταδίδεται ελεύθερα σε ανοικτά και μη ασφαλή κανάλια). Η πρόβλεψη της κλειδοροής μπορεί να είναι πιο εύκολα εφικτή αν ήδη γνωρίζουμε ένα μικρό τμήμα αυτής, το οποίο για τους αλγορίθμους ροής ταυτίζεται ουσιαστικά με το να γνωρίζουμε ένα μικρό τμήμα του αρχικού κρυφού μηνύματος (κάτι το οποίο στην πράξη δεν μπορεί να αποκλειστεί). Οποιαδήποτε τεχνική επίθεσης βασίζεται στην υπόθεση ότι ο υποκλοπέας γνωρίζει ένα τμήμα του αρχικού μηνύματος, ανήκει στην κατηγορία των επιθέσεων γνωστού κειμένου (known plaintext attacks).

Ο Shannon θεμελίωσε τις βασικές αρχές κρυπτογραφίας, όρισε την έννοια της απεριόριστης ασφάλειας και απέδειξε ότι ένας κρυπτογραφικός αλγόριθμος είναι απεριόριστα ασφαλής μόνο όταν η γνώση του κρυπτοκειμένου δεν παρέχει καμία απολύτως πληροφορία για το αρχικό μήνυμα [29]. Σε όρους χαρακτηριστικών της κλειδοροής αυτό μεταφράζεται στο ότι η κλειδοροή με την οποία κρυπτογραφούμε το μήνυμα πρέπει να είναι απολύτως τυχαία, αλλά και να έχει άπειρη περίοδο – πρακτικά, όση και το μέγεθος του αρχικού μηνύματος, έτσι ώστε να μην υπάρχει επαναληπτικότητα [12]. Παρόλο που τα χαρακτηριστικά αυτά παρέχουν απεριόριστη ασφάλεια, είναι πρακτικά αδύνατο να υπάρχουν στις ακολουθίες που χρησιμοποιούμε. Πρώτον, γιατί η παραγωγή μιας αληθινά τυχαίας σειράς ψηφίων είναι τεχνικά αδύνατη όταν αυτή παράγεται, τελικά, από ένα οποιασδήποτε μορφής υπολογιστικό (και, άρα, ντετερμινιστικό) σύστημα, και δεύτερον, γιατί η χρησιμοποίηση κλειδιού με μέγεθος ίσο με το μήνυμα παρουσιάζει δυσκολίες στην ασφαλή ανταλλαγή του. Προσπαθούμε λοιπόν να διασφαλίσουμε ότι η κλειδοροή πλησιάζει όσο περισσότερο γίνεται στα χαρακτηριστικά αυτά (της τυχαιότητας και του μεγέθους της περιόδου).

3.1.2 Γραμμικοί καταχωρητές ολίσθησης με ανάδραση (LFSR)

Οι γραμμικοί καταχωρητές ολίσθησης με ανάδραση (LFSR) είναι γεννήτριες κλειδοροών, οι οποίες παράγουν ψευδοτυχαίες ακολουθίες ψηφίων. Είναι πολύ δημοφιλείς γιατί παράγουν ψευδοτυχαίες ακολουθίες με πολύ μεγάλη περίοδο, ενώ ταυτόχρονα ικανοποιούν απαιτήσεις υψηλής ταχύτητας, χαμηλής κατανάλωσης ισχύος με εύκολη υλοποίηση στο υλικό (hardware). Η γεννήτρια αυτή αποτελείται από μια σειρά ψηφιακών βαθμίδων (θέσεων μνήμης, στις οποίες μπορούν να αποθηκευτούν οι δυαδικές τιμές 0 ή 1) και μια πύλη XOR η οποία έχει ως εισόδους κάποιες από τις ψηφιακές βαθμίδες (ανάδραση του LFSR). Ο LFSR μεταβάλλεται κάθε χρονική στιγμή και παράγει το επόμενο ψηφίο της ακολουθίας, με βάση την αρχική του κατάσταση και τις βαθμίδες ανάδρασης με την πύλη XOR. Η αρχική κατάσταση ενός LFSR αποτελεί και το μυστικό κλειδί της γεννήτριας.



Σχήμα 3.3 – Σχηματική αναπαράσταση ενός LFSR με N = 5

Αν θεωρήσουμε για παράδειγμα ένα LFSR μήκους N = 5, και ανάδραση στην πρώτη και τέταρτη βαθμίδα, από δεξιά (σχήμα 3.3), τότε η κλειδοροή που θα παραχθεί θα είναι η

```
0 1 1 0 0 0 1 1 1 1 1 0 0 1 1 0 1 0 0 1 0 0 0 0 1 0 1 0 1 1 1
```

η οποία έχει περίοδο 31, και θα επαναλαμβάνεται περιοδικά, ανάλογα με το μέγεθος του μηνύματος που θέλουμε να κρυπτογραφήσουμε.

Οι LFSR μπορούν να παράγουν ακολουθίες με μέγιστη περίοδο $2^N - 1$, όπου N είναι το μήκος του LFSR. Το ακριβές μήκος της περιόδου, δηλαδή αν θα είναι η μέγιστη δυνατή ή πιο μικρή, εξαρτάται από την ανάδραση του κάθε LFSR και την αρχική του κατάσταση.

Όσοι LFSR παράγουν ακολουθίες με τη μέγιστη δυνατή περίοδο ($2^N - 1$), ονομάζονται **πρωταρχικοί** (primitive), και αποτελούν μια ειδική περίπτωση LFSR. Οι πρωταρχικοί LFSR μας ενδιαφέρουν πάρα πολύ, αφού εκτός από το ότι παράγουν ακολουθίες με τη μέγιστη δυνατή περίοδο παρουσιάζουν επίσης και πολύ καλά κριτήρια τυχαιότητας (έχει αποδειχθεί ότι οι ακολουθίες των πρωταρχικών LFSR ικανοποιούν πάντα τα κριτήρια τυχαιότητας του Golomb [12]).

Συνεπώς οι ακολουθίες που παράγονται από πρωταρχικούς LFSR φαίνεται κατ' αρχάς ότι είναι κατάλληλες για χρήση σε κρυπτογραφικούς αλγορίθμους ροής. Εν τούτοις αυτό δεν είναι ακριβές, όπως καταδεικνύουμε στη συνέχεια.

Γραμμική πολυπλοκότητα ακολουθίας

Γενικά, μια ακολουθία μπορεί να παραχθεί από πολλούς διαφορετικούς LFSR. Για παράδειγμα η ακολουθία 1011100 μπορεί να παραχθεί από τον LFSR με N=3, ανάδραση στις θέσεις 1 και 2 και αρχική κατάσταση 101, αλλά και από άλλους LFSR με μεγαλύτερο μέγεθος N.

Γραμμική πολυπλοκότητα μιας ακολουθίας ονομάζεται το μέγεθος του μικρότερου LFSR από τον οποίο μπορεί να παραχθεί η ακολουθία [42]. Στο παράδειγμα της ακολουθίας 1011100, ο LFSR που περιγράφηκε να την παράγει (N=3, ανάδραση στις θέσεις 1 και 2 και αρχική κατάσταση 101) είναι όντως ο μικρότερος LFSR που μπορεί να τη δημιουργήσει. Άρα η γραμμική πολυπλοκότητα της ακολουθίας 1011100 είναι 3.

Η γραμμική πολυπλοκότητα αποτελεί ένα από τα πιο σημαντικά χαρακτηριστικά της ακολουθίας, αφού, όπως θα δούμε και πιο κάτω, όσο πιο χαμηλή γραμμική πολυπλοκότητα έχει μια ακολουθία, τόσο πιο ευάλωτη είναι σε επιθέσεις γνωστού κειμένου. Άρα, η ακολουθία που χρησιμοποιούμε, πρέπει τελικά να έχει όσο πιο υψηλή γραμμική πολυπλοκότητα γίνεται.

Επιθέσεις γνωστού κειμένου

Παρόλο που η χρήση των πρωταρχικών LFSR δημιουργούν ακολουθίες με πολύ ελκυστικά πλεονεκτήματα, είναι ευάλωτοι σε επιθέσεις γνωστού κειμένου. Μπορεί δηλαδή κάποιος υποκλοπέας να **υπολογίσει το μυστικό κλειδί** (δηλαδή την αρχική κατάσταση του LFSR) αν γνωρίζει απλά ένα τμήμα του αρχικού – μη κρυπτογραφημένου – μηνύματος. Αφού ο

υποκλοπέας υπολογίσει την αρχική κατάσταση του LFSR, μπορεί να δημιουργήσει μόνος του την κλειδοροή και να προχωρήσει στην αποκρυπτογράφηση.

Παράδειγμα απλής επίθεσης γνωστού κειμένου

Έστω κρυπτογραφημένο μήνυμα: `c = 1100100011000010101010111…` ,

από LFSR N=5 με ανάδραση στις βαθμίδες 1 και 2 (από δεξιά) – άγνωστης αρχικής κατάστασης,

με γνωστά τα πρώτα 5 bit του αρχικού μηνύματος: `m = 01011`,

παράγουμε τα πρώτα 5 bit της ακολουθίας κρυπτογράφησης:

```
    1100100011000010101010111 κρυπτοκείμενο
XOR 01011                     γνωστό κείμενο
    --------------------------
    10010                     τα πρώτα 5 bit της κλειδοροής
```

Άρα η αρχική κατάσταση του LFSR είναι `10010`, αφού το μήκος του LFSR είναι N=5 και έχουμε τα 5 πρώτα bit της κλειδοροής.

Σχήμα 3.4 – Παράδειγμα απλής επίθεσης γνωστού κειμένου

Η επίθεση πραγματοποιείται όταν εκτελεστεί η πράξη XOR επάνω στο κρυπτοκείμενο και το γνωστό κείμενο, οπόταν και προκύπτει το κομμάτι της κλειδοροής το οποίο κρυπτογράφησε το αντίστοιχο κομμάτι του γνωστού κειμένου. Στην απλή μορφή της επίθεσης, αποκαλύπτεται με αυτό τον τρόπο αρχική κατάσταση του LFSR, άρα μπορεί ο επιτιθέμενος να παράξει ολόκληρη την ακολουθία με την οποία να αποκρυπτογραφήσει το κρυπτοκείμενο.

Πέρα όμως από την απλή μορφή της επίθεσης (όπου το γνωστό κείμενο βρίσκεται στην αρχή του κρυπτοκειμένου), ο επιτιθέμενος μπορεί να υπολογίσει τον LFSR κρυπτογράφησης και με οποιοδήποτε γνωστό κείμενο. Αυτό γίνεται κατορθωτό με την αξιοποίηση του **αλγόριθμου Berlekamp – Massey (BMA)**.

Ο αλγόριθμος Berlekamp – Massey λαμβάνει σαν είσοδο μια ακολουθία και υπολογίζει τον μικρότερο LFSR που την παράγει. Δίνει δηλαδή το μέγεθος (άρα τη γραμμική πολυπλοκότητα), και την ανάδραση του μικρότερου LFSR που παράγει τη δοθείσα ακολουθία.

Χρειάζεται όμως να δώσουμε στον BMA ακολουθία αρκετά μεγάλη έτσι ώστε ο LFSR που θα μας δώσει πίσω να είναι και ο μοναδικός που παράγει την ακολουθία. Αν δώσουμε πολύ μικρή ακολουθία, τότε το αποτέλεσμα του BMA δεν θα είναι ένας LFSR μοναδικός για τη δοθείσα ακολουθία και έτσι δεν θα μπορούμε χρησιμοποιήσουμε το αποτέλεσμα για να παράξουμε και την υπόλοιπη ακολουθία.

$b \leftarrow 1$
$k \leftarrow 1$
$B(x) \leftarrow 1$
$n \leftarrow 0$
$L \leftarrow 0$ % linear complexity
$c(x) \leftarrow 1$ % feedback polynomial
while $n < N$ **do**
  $d \leftarrow y_n + \sum_{i=1}^{L} c_i y_{n-i}$
  if $d \neq 0$ **then**
    if $2L > N$ **then** % the linear complexity does not increase
    $c(x) \leftarrow c(x) - db^{-1}x^k B(x)$
    $k \leftarrow k + 1$
    elseif $L \leq \frac{n}{2}$ **then** % the linear complexity increases
      $T(x) \leftarrow c(x)$
      $c(x) \leftarrow c(x) - db^{-1}x^k B(x)$
      $L \leftarrow n + 1 - L$ % new value of complexity
      $b \leftarrow d$
      $B(x) \leftarrow T(x)$
      $k \leftarrow 1$
    endif
  else % $d = 0$, i.e. no change of LFSR
    $x \leftarrow x + 1$
  endif
  $n \leftarrow n + 1$
endwhile

Σχήμα 3.5 – Ο αλγόριθμος Berlekamp – Massey [43]

Το πόσο μεγάλη θα πρέπει να είναι η γνωστή ακολουθία που θα δώσουμε στον BMA, για να λάβουμε κάποιο μοναδικό και χρησιμοποιήσιμο LFSR εξαρτάται απόλυτα από τη γραμμική πολυπλοκότητα της αρχικής ακολουθίας που χρησιμοποιήθηκε κατά την κρυπτογράφηση. Χρειάζεται να παρέχουμε στον BMA γνωστή ακολουθία μήκους τουλάχιστον 2L διαδοχικών ψηφίων, όπου L είναι η γραμμική πολυπλοκότητα της αρχικής ακολουθίας που ψάχνουμε.

Άρα, **η γραμμική πολυπλοκότητα της ακολουθίας που θα χρησιμοποιήσουμε καθορίζει άμεσα πόσο εύκολα ή δύσκολα θα μπορέσει κάποιος υποκλοπέας να εκτελέσει επίθεση**

γνωστού κειμένου στο κρυπτοκείμενό μας. Όσο πιο μεγάλη είναι η γραμμική πολυπλοκότητα της ακολουθίας μας, τόσο πιο μεγάλο θα πρέπει να είναι το κομμάτι του αρχικού κειμένου που θα χρειαστεί να γνωρίζει ο υποκλοπέας [43]. Πρέπει να σημειωθεί ότι, ακόμα και αν δεν χρησιμοποιείται ένας απλός LFSR για την παραγωγή της κλειδοροής αλλά κάποιος πιο σύνθετος τρόπος παραγωγής της, πάλι υπάρχει το ενδεχόμενο η κλειδοροή να μην έχει υψηλή γραμμική πολυπλοκότητα και, συνεπώς, λόγω του αλγορίθμου Berlekamp-Massey να είναι προβλέψιμη. Συνεπώς, για την επίτευξη υψηλής γραμμικής πολυπλοκότητας δεν αρκεί απλά να μην χρησιμοποιούμε LFSR ως μοναδικό δομικό συστατικό για γεννήτρια κλειδοροής.

3.2 Αλγόριθμοι τμήματος

Μια άλλη σημαντική ομάδα αλγορίθμων συμμετρικού κλειδιού είναι οι αλγόριθμοι τμήματος (Block Cyphers). Σε αντίθεση με τους αλγόριθμους ροής, στους αλγορίθμους τμήματος το αρχικό κείμενο χωρίζεται σε τμήματα (blocks) σταθερού μεγέθους, τα οποία και κρυπτογραφούνται ξεχωριστά το ένα από το άλλο. Η αποκρυπτογράφηση από τον παραλήπτη γίνεται επίσης κατά τμήμα.

Ο οργανισμός προτυποποίησης NIST καθιέρωσε ως πρότυπο αλγόριθμο τμήματος τον αλγόριθμο AES (Advanced Encryption Standard), ο οποίος αντικατέστησε τον παλαιότερο DES (Data Encryption Standard). Σημειώνεται ότι, για την κατηγορία αλγορίθμων ροής που μελετήθηκαν νωρίτερα, δεν έχει οριστεί αντίστοιχα κάποιος πρότυπος αλγόριθμος κρυπτογράφησης.

Οι αλγόριθμοι τμήματος, με κατάλληλο τρόπο λειτουργίας, μπορούν να χρησιμοποιηθούν ως αλγόριθμοι ροής – συγκεκριμένα, σε αυτήν την περίπτωση, ο ίδιος ο αλγόριθμος τμήματος αποτελεί, με κατάλληλες εισόδους και αρχική κατάσταση, τη γεννήτρια της κλειδοροής [30].

3.3 Αλγόριθμοι Δημόσιου κλειδιού

Οι αλγόριθμοι δημοσίου κλειδιού (public key) χρησιμοποιούν δύο διαφορετικά κλειδιά, ένα κατά την κρυπτογράφηση και ένα κατά την αποκρυπτογράφηση. Τα κλειδιά αυτά έχουν σχέση μεταξύ τους τέτοια που ενώ το ένα κωδικοποιεί και το άλλο αποκωδικοποιεί, είναι μαθηματικά αδύνατο κάποιος που γνωρίζει το ένα κλειδί να μπορεί να προσδιορίσει το άλλο. Τα δύο κλειδιά ονομάζονται Δημόσιο κλειδί και Ιδιωτικό κλειδί και χρησιμοποιούνται ως εξής:

- **Δημόσιο κλειδί (public key)**. Είναι δημοσιευμένο και γνωστό σε όλους. Ο αποστολέας ενός μηνύματος χρησιμοποιεί το δημόσιο κλειδί του παραλήπτη για να το κρυπτογραφήσει. Δεν τίθεται κάποιο θέμα ασφαλούς ανταλλαγής κλειδιών, αφού το κλειδί αυτό είναι ήδη δημοσιευμένο και γνωστό σε όλους.

  Το κρυπτογραφημένο με το δημόσιο κλειδί κείμενο, αποκρυπτογραφείται μόνο με τη χρήση του ιδιωτικού κλειδιού.

- **Ιδιωτικό κλειδί (private key)**. Είναι το δεύτερο κλειδί του ζεύγους και είναι μυστικό. Κανείς δεν πρέπει να γνωρίζει το ιδιωτικό κλειδί, παρά μόνο ο ιδιοκτήτης του. Χρησιμοποιείται για την αποκωδικοποίηση μηνυμάτων που κωδικοποιήθηκαν με το αντίστοιχο δημόσιο κλειδί. Αφού δεν γνωρίζει κανείς το ιδιωτικό κλειδί, μόνο ο νόμιμος παραλήπτης του μηνύματος μπορεί να το αποκρυπτογραφήσει.

Εκτός από την κρυπτογράφηση μηνυμάτων, χρησιμοποιούμε το δημόσιο κλειδί και για τη δημιουργία **ψηφιακών υπογραφών**, για την αυθεντικοποίηση των μηνυμάτων. Η διαδικασία υπογραφής ενός μηνύματος είναι να κωδικοποιηθεί το αποτύπωμα του από τον αποστολέα χρησιμοποιώντας το δικό του ιδιωτικό κλειδί. Αυτό σημαίνει ότι το κρυπτογραφημένο αυτό αποτύπωμα μπορεί τώρα να αποκρυπτογραφηθεί μόνο με το δημόσιο κλειδί του αποστολέα. Μπορεί έτσι ο παραλήπτης να επιβεβαιώσει ότι το αποτύπωμα κρυπτογραφήθηκε από το συγκεκριμένο αποστολέα αφού χρησιμοποιήσει το δημόσιο κλειδί του αποστολέα. Κωδικοποιημένο μήνυμα το οποίο αποκωδικοποιείται ορθά με το δημόσιο κλειδί κάποιου, σημαίνει ότι κρυπτογραφήθηκε με τη χρήση του αντίστοιχου ιδιωτικού κλειδιού.

3.4 Συνδυασμός συμμετρικών αλγορίθμων με δημοσίου κλειδιού

Για να γίνει μια ασφαλής ανταλλαγή μηνυμάτων σήμερα, χρησιμοποιείται συνδυασμός συμμετρικών αλγορίθμων με δημοσίου κλειδιού. Το μήνυμα το οποίο χρειάζεται να αποσταλεί κρυπτογραφείται με κάποιο συμμετρικό αλγόριθμο, ενώ το συμμετρικό κλειδί που χρησιμοποιήθηκε αποστέλλεται στον παραλήπτη του μηνύματος κρυπτογραφημένο με αλγόριθμο δημοσίου κλειδιού. Αλγόριθμος δημοσίου κλειδιού μπορεί να χρησιμοποιηθεί και για τη δημιουργία ψηφιακής υπογραφής του αρχικού μηνύματος, πριν από την κρυπτογράφηση του. Ο παραλήπτης του μηνύματος λαμβάνει δύο κρυπτογραφημένα μηνύματα:

- Το συμμετρικό κλειδί που χρησιμοποιήθηκε για την κρυπτογράφηση του αρχικού μηνύματος, κρυπτογραφημένο με το δημόσιο του κλειδί (του παραλήπτη). Χρησιμοποιεί το ιδιωτικό του κλειδί για να το αποκρυπτογραφήσει.

- Το αρχικό κείμενο, κρυπτογραφημένο με συμμετρικό αλγόριθμο. Χρησιμοποιεί το συμμετρικό κλειδί (βλ. προηγούμενο σημείο) για να το αποκρυπτογραφήσει. Στην περίπτωση που το αρχικό κείμενο είναι ψηφιακά υπογεγραμμένο, χρησιμοποιείται το δημόσιο κλειδί του αποστολέα για να πιστοποιηθεί η υπογραφή.

Με αυτό τον τρόπο συνδυάζονται τα θετικά των δύο ειδών αλγορίθμων, «απαλύνοντας» κατά το δυνατόν τα μειονεκτήματά τους. Το κυρίως μήνυμα κρυπτογραφείται με συμμετρικό αλγόριθμο, ο οποίος είναι πολύ πιο γρήγορος και μη κοστοβόρος σε υπολογιστική ισχύ από αυτούς του δημόσιου κλειδιού, ενώ ξεπερνούμε το πρόβλημα ανταλλαγής του συμμετρικού κλειδιού με τη χρήση αλγόριθμου δημοσίου κλειδιού: κρυπτογραφούμε το συμμετρικό κλειδί με δημόσιο κλειδί και το αποστέλλουμε στον παραλήπτη του μηνύματος μαζί με το συμμετρικά κρυπτογραφημένο μήνυμα.

Κεφάλαιο 4
Ασφάλεια Συνθηματικών

Η χρήση συνθηματικών για την εξουσιοδοτημένη πρόσβαση σε ένα σύστημα αποτελεί την πρώτη γραμμή άμυνας έναντι των κακόβουλων εισβολέων [30]. Τα περισσότερα συστήματα και εφαρμογές ζητούν από τους χρήστες να παρέχουν κάποιο όνομα (αναγνωριστικό) μαζί με ένα συνθηματικό. Όταν κάποιος χρήστης εισάγει το σωστό ζεύγος ονόματος-συνθηματικού, τότε η εφαρμογή / σύστημα επιτρέπει πρόσβαση σε αυτό το χρήστη. Ο εισβολέας λοιπόν, επιχειρεί να αποκτήσει κάποιο συνθηματικό για να αποκτήσει πρόσβαση στο σύστημα σαν πρώτο βήμα μιας ευρύτερης επίθεσης.

4.1 Υποκλοπή μεμονωμένων συνθηματικών σε ένα σύστημα

Στη σχετική βιβλιογραφία [28], [30], αλλά και σε μελέτες όπως αυτή της A. Alvares, "How Crackers Crack Passwords or What Passwords to Avoid" [01], παρουσιάζονται οι πιο δημοφιλείς μέθοδοι που χρησιμοποιούν οι κακόβουλοι εισβολείς για να αποκτήσουν μεμονωμένα συνθηματικά κάποιου συστήματος:

- Δοκιμάζουν να **μαντέψουν κάποιο συνθηματικό** με διάφορους τρόπους, όπως να δοκιμάσουν τα αρχικά "εργοστασιακά" συνθηματικά του συστήματος (για παράδειγμα, ο διαχειριστής κάποιας ιστοσελίδας μπορεί να μην αλλάξει το αρχικό συνθηματικό πρόσβασης στον διακομιστή Apache), ή να δοκιμάσουν όλους τους συνδυασμούς χαρακτήρων για πολύ μικρά συνθηματικά ή δοκιμάζουν όλες τις λέξεις σε έτοιμα λεξικά πιθανών συνθηματικών, ή να δοκιμάσουν όλους τους πιθανούς αριθμούς πινακίδων αυτοκινήτων, κλπ.

- Μαζεύουν **προσωπικές πληροφορίες για κάποιο χρήστη** (αριθμό τηλεφώνου, διεύθυνση, ονόματα παιδιών) και συντάσσουν λεξικό πιθανών συνθηματικών προσανατολισμένα στο συγκεκριμένο χρήστη ως πιθανά για να τα έχει επιλέξει. Στη συνέχεια δοκιμάζουν όλες τις λέξεις του λεξικού αυτού.

- Υποκλέπτουν το συνθηματικό **κατευθείαν από το χρήστη**, με την εγκατάσταση κάποιου κακόβουλου λογισμικού (ιού) στον υπολογιστή του ή με άλλους τρόπους (π.χ. μέσω παρακολούθησης του δικτύου του, μέσω παραπλανητικών μηνυμάτων κτλ).

Για την αντιμετώπιση των πάνω μεθόδων απόκτησης ή και υποκλοπής συνθηματικών, τα συστήματα χρειάζεται να υπόκεινται σε συγκεκριμένα μέτρα και να υλοποιούν μεθόδους αντιμετώπισης. Επίσης, οι χρήστες των συστημάτων οφείλουν να ακολουθούν συγκεκριμένες καλές πρακτικές. Μερικές από αυτές προτείνονται από σημαντικούς οργανισμούς όπως τους CERT ([10], [11]), SANS ([27]) και OWASP ([16]) αλλά αναφέρονται και στη βιβλιογραφία [30]:

- Απαγόρευση πολλών διαδοχικών δοκιμών σύνδεσης, σε περίπτωση αποτυχίας. Το σύστημα να μην επιτρέπει πολλές συνεχόμενες αποτυχημένες απόπειρες σύνδεσης.

- Το συνθηματικό κάθε χρήστη πρέπει να έχει βαθμό πολυπλοκότητας τέτοιο ώστε να το καθιστά δύσκολα προβλέψιμο. Μπορεί να υιοθετηθούν κανόνες για το μέγεθός του (πάνω από 8 χαρακτήρες) αλλά και για το είδος των χαρακτήρων που το αποτελούν (για παράδειγμα να περιλαμβάνει χαρακτήρες του αλφαβήτου σε κεφαλαία και μικρά γράμματα, αριθμητικά ψηφία αλλά και ειδικούς χαρακτήρες όπως η τελεία, κόμμα, θαυμαστικό κλπ).

- Το συνθηματικό κάθε χρήστη να έχει μικρή "διάρκεια ζωής". Κάθε μικρό χρονικό διάστημα να απαιτείται από το χρήστη να το αλλάζει, με περιορισμό να μην μπορεί να επαναλάβει συνθηματικό που χρησιμοποίησε ο ίδιος χρήστης στο παρελθόν.

- Να μην επιτρέπεται να περιλαμβάνεται στο συνθηματικό το όνομα του χρήστη, ή λέξεις από κάποιο λεξικό εύκολων συνθηματικών.

- Απαγορεύεται οι χρήστες να τηρούν αντίγραφο του συνθηματικού τους σε χαρτί, και ειδικά σε σημεία κοντά ή και πάνω στον υπολογιστή τους.

- Απαγορεύεται στους χρήστες να αποκαλύπτουν το συνθηματικό τους σε οποιονδήποτε, ούτε ακόμη και στο διευθυντή τους ή σε συναδέλφους τους. Επίσης απαγορεύεται στους χρήστες του συστήματος να ζητούν από άλλους χρήστες να τους αποκαλύπτουν τα συνθηματικά τους.

Οι οργανισμοί που δημιουργούν, συντηρούν και διαχειρίζονται εφαρμογές και συστήματα κωδικοποιούν τα μέτρα αυτά στην ***πολιτική ασφάλειας***, η οποία και επιβάλλεται σε όλους τους χρήστες των συστημάτων και εφαρμογών.

4.2 Υποκλοπή του αρχείου συνθηματικών

Για να μπορούν οι εφαρμογές να πραγματοποιήσουν την αυθεντικοποίηση των χρηστών τους, διατηρούν κατάλογο με τα αποδεκτά ζεύγη ονόματος και συνθηματικού. Αυτός ο κατάλογος διατηρείται σε κάποιο αρχείο συνθηματικών, συνήθως σε μορφή βάσης δεδομένων. Επιπρόσθετα λοιπόν από τις μεθόδους αποκάλυψης μεμονωμένων συνθηματικών, οι οποίες συζητήθηκαν πιο πάνω, οι εισβολείς συστημάτων επιχειρούν συχνά επιθέσεις για να αποκτήσουν το σύνολο των συνθηματικών, βάζοντας στο στόχαστρο τους το **αρχείο συνθηματικών**.

Για να προστατευθεί το αρχείο αυτό, μπορούν να υιοθετηθούν δύο μέτρα ασφαλείας:

- **Έλεγχος προσπέλασης (access control)**. Το σύστημα υλοποιεί μηχανισμούς με τους οποίους ελέγχει και περιορίζει την προσπέλαση στο αρχείο συνθηματικών. Γίνεται προσπάθεια έτσι να προστατευθεί το αρχείο από μη εξουσιοδοτημένη πρόσβαση στα δεδομένα του.

- **Υλοποίηση συνάρτησης κατακερματισμού**. Στο αρχείο αποθηκεύεται το αποτύπωμα των συνθηματικών, και όχι τα συνθηματικά αυτούσια. Κατά τη διαδικασία αυθεντικοποίησης κάποιου χρήστη, το σύστημα υπολογίζει την κατακερματισμένη τιμή (αποτύπωμα) του συνθηματικού που δίνεται από το χρήστη, και στη συνέχεια συγκρίνει το αποτύπωμα που προκύπτει με αυτό που τηρείται, για το συγκεκριμένο χρήστη, στο αρχείο συνθηματικών.

  Σε περίπτωση που κάποιος επιτιθέμενος αποκτήσει πρόσβαση στο αρχείο δεδομένων, δεν θα μπορέσει να έχει τα αυτούσια συνθηματικά, αφού η συνάρτηση κατακερματισμού είναι μιας κατεύθυνσης, δηλαδή μη αντιστρέψιμη, άρα δεν μπορεί κάποιος να εξάγει το αυτούσιο συνθηματικό χρησιμοποιώντας το αποτύπωμα. Περισσότερα για τις συναρτήσεις κατακερματισμού ακολουθούν στη συνέχεια.

Στις περιπτώσεις που ο επιτιθέμενος επιτύχει να προσπεράσει τα μέτρα ελέγχου προσπέλασης, και κερδίσει τελικά πρόσβαση στο αρχείο συνθηματικών, τότε απομονώνει κάποιο αντίγραφό του εκτός δικτύου και πραγματοποιεί επάνω του επιθέσεις, με σκοπό να καταφέρει να βρει τα συνθηματικά που αντιστοιχούν στα αναγνωριστικά που περιλαμβάνονται στο αρχείο αυτό [30].

Επιθέσεις λεξικού

Ο επιτιθέμενος συγκρίνει τα αποτυπώματα συνθηματικών που περιλαμβάνονται σε λεξικό συνθηματικών με αυτά του αρχείου συνθηματικών. Αν κάποιο αποτύπωμα ταιριάζει, τότε μπορεί να χρησιμοποιήσει αυτό το συνθηματικό μαζί με το αντίστοιχο αναγνωριστικό για να κερδίσει πρόσβαση στην εφαρμογή. Για να αντιμετωπίσει κάποιος αυτό τον κίνδυνο θα πρέπει να έχει εγκατεστημένο σύστημα ανίχνευσης εισβολών, το οποίο να ειδοποιήσει το διαχειριστή για την υποκλοπή του αρχείο συνθηματικών και έτσι να προχωρήσει στην άμεση αντικατάσταση των συνθηματικών.

Επίθεση δημοφιλούς συνθηματικού

Ο επιτιθέμενος δεν ασχολείται με τα αποτυπώματα συνθηματικών που υπάρχουν στο αρχείο. Αντί αυτού, απομονώνει μόνο τα αναγνωριστικά και στη συνέχεια δοκιμάζει να συνδεθεί στην εφαρμογή χρησιμοποιώντας ένα ένα όλα αυτά τα αναγνωριστικά μαζί με κάποιο πολύ δημοφιλές συνθηματικό. Χρησιμοποιεί το ίδιο συνθηματικό για όλα τα αναγνωριστικά, με την ελπίδα ότι όλο και κάποιος χρήστης θα το επέλεξε για συνθηματικό του. Για να αντιμετωπισθεί ο

κίνδυνος αυτής της επίθεσης, θα πρέπει το σύστημα να παρακολουθεί όλες τις απόπειρες σύνδεσης μαζί με τη διεύθυνση IP από την οποία εκτελούνται. Το σύστημα θα πρέπει να απαγορεύει πολλαπλές απόπειρες σύνδεσης, με διαφορετικά συνθηματικά από την ίδια διεύθυνση διαδικτύου (IP address).

Από τα παραπάνω προκύπτει ότι δεν πρέπει οι χρήστες να επαφίενται μόνο στα μέτρα ασφαλείας του συστήματος για την προστασία των συνθηματικών τους: θα πρέπει σε κάθε περίπτωση να επιλέγουν συνθηματικά που να μην μπορούν να ανακαλυφθούν μέσω των προαναφερθεισών τεχνικών. Δυστυχώς, στην πράξη, οι χρήστες επιμένουν να επιλέγουν συνθηματικά εύκολα προς απομνημόνευση και εξαιρετικά προβλέψιμα.

4.3 Συναρτήσεις κατακερματισμού (hash functions)

Μια συνάρτηση κατακερματισμού δέχεται σαν είσοδο οποιουδήποτε μεγέθους μήνυμα και επιστρέφει μια έξοδο σταθερού πάντα μεγέθους. Η έξοδος της συνάρτησης κατακερματισμού ονομάζεται ***αποτύπωμα του μηνύματος*** που δώσαμε στην είσοδο. Το αποτύπωμα που δίνεται είναι πάντα σταθερού μεγέθους, ανεξαρτήτως του μεγέθους του αρχικού μηνύματος στην είσοδο της συνάρτησης. Για μια καλή συνάρτηση κατακερματισμού αυτό καθιστά το αποτύπωμα μη αναστρέψιμο, δηλαδή δεν μπορεί να ανακτηθεί το αρχικό μήνυμα από το αποτύπωμα. Από την άλλη, το αποτύπωμα είναι (πρακτικά) μοναδικό για το αρχικό μήνυμα, δηλαδή δεν μπορεί να βρεθούν δύο διαφορετικά αρχικά μηνύματα τα οποία να δίνουν το ίδιο αποτύπωμα. Μερικοί γνωστοί αλγόριθμοι κατακερματισμού είναι οι SHA-1, SHA-2, MD5, κλπ., ενώ από το 2012 έχει επιλεγεί, ως πρότυπος αλγόριθμος κατακερματισμού, ο Keccak (γνωστός και ως SHA-3).

Οι συναρτήσεις κατακερματισμού χρησιμοποιούνται για την ασφαλή αποθήκευση συνθηματικών, αλλά και στην κρυπτογραφία (πιστοποίηση ακεραιότητας δεδομένων και αυθεντικοποίηση μηνύματος).

Σε αυτό το τμήμα θα ασχοληθούμε με τη χρήση των συναρτήσεων κατακερματισμού για την ασφαλή αποθήκευση συνθηματικών. Η μέθοδος αυτή χρησιμοποιείται κατά κανόνα στα λειτουργικά συστήματα UNIX και Windows, καθώς και σε άλλα λειτουργικά συστήματα. Η ιδέα είναι ότι δεν αποθηκεύεται στο σύστημα κανένα συνθηματικό σε μορφή απλού κειμένου, αλλά αποθηκεύεται μόνο το αποτύπωμά του. Έτσι, σε περίπτωση που κάποιος επιτιθέμενος

καταφέρει να αποκτήσει πρόσβαση στο αρχείο συνθηματικών, να μην έχει στα χέρια του τα αυτούσια συνθηματικά.

Αποθήκευση αποτυπώματος συνθηματικού – Δημιουργία νέου χρήστη

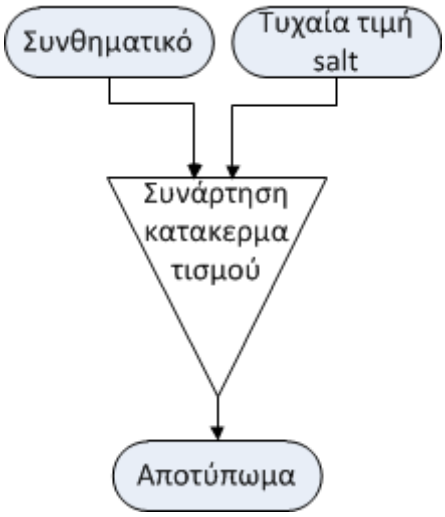
Για να προσθέσουμε τα στοιχεία αυθεντικοποίησης κάποιου νέου χρήστη στο αρχείο συνθηματικών, χρειαζόμαστε τρεις τιμές:

- Το αναγνωριστικό (username) του χρήστη. Αυτό προέρχεται συνήθως από το χρήστη και χρειάζεται να είναι μοναδικό στο σύστημα.

- Το συνθηματικό του χρήστη, το οποίο προέρχεται από το χρήστη.

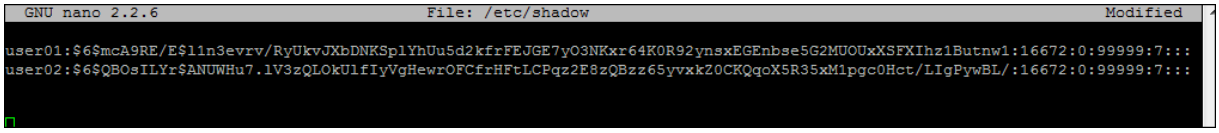
- Μια τυχαία τιμή (salt) η οποία δημιουργείται από το σύστημα για κάθε νέο χρήστη.

Το σύστημα ενώνει την τιμή salt μαζί με το συνθηματικό του χρήστη, και χρησιμοποιεί συνάρτηση κατακερματισμού για να δημιουργήσει το αποτύπωμα του συνδυασμού των δύο (salt και συνθηματικού). Στη συνέχεια, αποθηκεύονται στο αρχείο συνθηματικών το αναγνωριστικό χρήστη (username), η τυχαία τιμή (salt) σε αυτούσια μορφή απλού κειμένου και το αποτύπωμα που προέκυψε από τη συνάρτηση κατακερματισμού.



Σχήμα 4.1 – Το αποτύπωμα προκύπτει αφού δώσουμε στη συνάρτηση κατακερματισμού το συνθηματικό και την τυχαία τιμή salt

Για να πειραματιστούμε με την επίδραση του salt στη δημιουργία του αποτυπώματος κατά τη δημιουργία νέου χρήστη, δημιουργήσαμε σε λειτουργικό σύστημα UNIX (Debian) δύο νέους χρήστες (user01 και user02) και δώσαμε και στους δύο χρήστες το ίδιο συνθηματικό. Όταν επιθεωρήσαμε το αρχείο συνθηματικών του συστήματος (/etc/shadow) οι δύο χρήστες εμφανίζονται με διαφορετικά αποτυπώματα αποθηκευμένα (σχήμα 4.2). Η διαφορά αποδίδεται στην επίδραση του salt, το οποίο είναι διαφορετικό για τον κάθε χρήστη.



Σχήμα 4.2 – Το αρχείο /etc/shadow
το οποίο δείχνει τους χρήστες user01 και user02
με το αποτύπωμα του συνδυασμού συνθηματικού και salt

Η χρήση της τυχαίας τιμής salt στη δημιουργία του αποτυπώματος εξυπηρετεί τρεις σκοπούς [30]:

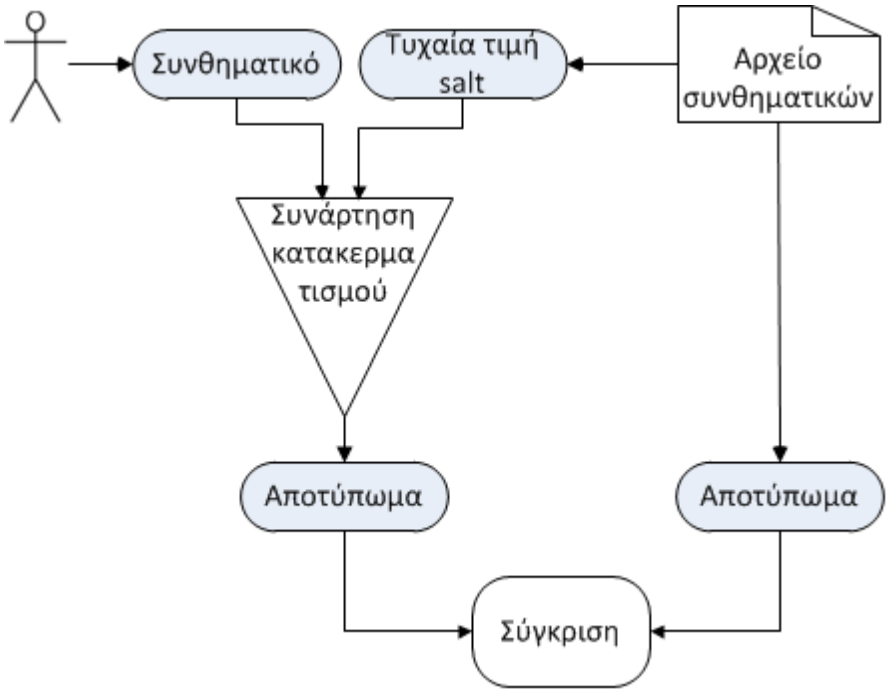
α) Δεν μπορεί κάποιος να καταλάβει αν υπάρχουν διπλά συνθηματικά στο αρχείο. Αν δύο χρήστες επιλέξουν τυχαία να έχουν το ίδιο συνθηματικό, αυτά δεν θα δώσουν το ίδιο αποτύπωμα, αφού στη συνάρτηση κατακερματισμού θα εισαχθεί διαφορετικό salt για κάθε χρήστη.

β) Παρόμοια, αν κάποιος χρήστης χρησιμοποιήσει το ίδιο συνθηματικό σε δύο ή περισσότερες εφαρμογές, αυτές οι εφαρμογές δεν θα έχουν αποθηκευμένο το ίδιο αποτύπωμα, αφού η κάθε μια θα δημιουργήσει το δικό της, διαφορετικό, salt για το χρήστη.

γ) Αυξάνει πάρα πολύ το χρόνο που χρειάζεται κάποιος να εκτελέσει επίθεση λεξικού στο αρχείο συνθηματικών. Χωρίς τη χρήση salt, η επίθεση λεξικού πραγματοποιείται με τη σύγκριση των αποτυπωμάτων του λεξικού με αυτά του αρχείου συνθηματικών. Με τη χρήση salt, χρειάζεται για κάθε λέξη στο λεξικό να παράγεται ξεχωριστό αποτύπωμα για κάθε ένα salt που βρίσκεται στο αρχείο συνθηματικών. Αυτό αυξάνει παραγοντικά το χρόνο και κόπο που χρειάζεται κάποιος για να εκτελέσει την επίθεση λεξικού.

Αυθεντικοποίηση του χρήστη

Όταν κάποιος χρήστης του συστήματος χρειαστεί να ζητήσει πρόσβαση σε αυτό, τότε παρέχει στο σύστημα το αναγνωριστικό του (username) και το συνθηματικό του (σε μορφή απλού κειμένου). Το σύστημα λαμβάνει τα στοιχεία αυτά και χρησιμοποιώντας το username βρίσκει από το αρχείο συνθηματικών το αντίστοιχο salt του χρήστη αυτού. Στη συνέχεια συνενώνει το salt με το συνθηματικό που δόθηκε από το χρήστη και παράγει το αντίστοιχο αποτύπωμα, το οποίο συγκρίνεται με αυτό που υπάρχει στο αρχείο συνθηματικών και ανάλογα παρέχεται πρόσβαση ή όχι.



Σχήμα 4.3 – Η διαδικασία αυθεντικοποίησης με τη χρήση salt

Κεφάλαιο 5
Διαδικτυακές επιθέσεις "έγχυσης" παράνομου κώδικα

Στο κεφάλαιο αυτό θα περιγράψουμε δύο από τις πλέον σημαντικές επιθέσεις που μπορούν να λάβουν χώρα σε περιβάλλον διαδικτυακών εφαρμογών: τις επιθέσεις "έγχυσης" κώδικα SQL, (SQL injection) και τις επιθέσεις εισαγωγής κώδικα μέσω ιστοσελίδων (cross site scripting ή XSS). Και στις δύο περιπτώσεις απώτερος στόχος είναι η "υπόγεια" εκτέλεση κακόβουλου κώδικα: στην πρώτη περίπτωση ο κώδικας εκτελείται στην υποκείμενη βάση δεδομένων, της οποίας διεπαφή είναι μία διαδικτυακή εφαρμογή, ενώ στη δεύτερη περίπτωση ο κώδικας εκτελείται στον υπολογιστή του απλού χρήστη που πλοηγείται σε μια ιστοσελίδα. Παρόλο που είναι πλέον πολύ γνωστές οι συγκεκριμένες επιθέσεις, με γνωστά εργαλεία αντιμετώπισής τους, εν τούτοις μέχρι και σήμερα εξακολουθούν να αποτελούν πολύ μεγάλο κίνδυνο λόγω της ευρείας εφαρμογής τους και των συνεπειών που επιφέρουν.

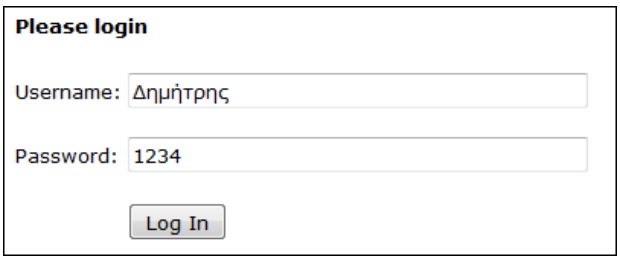
5.1 Επιθέσεις τύπου "SQL injection"

Οι επιθέσεις έγχυσης SQL (SQL injection) αποτελούν τη μεγαλύτερη απειλή για διαδικτυακές εφαρμογές, με μεγάλες εταιρίες όπως οι Sony Pictures, PBS, MySQL.com και Yahoo [33] να έχουν πέσει θύματα αυτής της επίθεσης στο παρελθόν. Χαρακτηριστικό παράδειγμα στην ελληνική πραγματικότητα είναι η περίπτωση του ελληνικού ιστοτόπου της Sony Music που υπήρξε θύμα αυτής της επίθεσης το 2011 [39], με συνέπεια να διαρρεύσουν τα προσωπικά δεδομένα των εγγεγραμμένων χρηστών – μάλιστα, τα συνθηματικά αυτών τηρούνται σε αυτούσια μορφή και όχι σε μορφή αποτυπώματος (βλ. προηγούμενο κεφάλαιο), οπότε και οι συνέπειες της επίθεσης κρίθηκαν εξαιρετικά σημαντικές (αν αναλογιστεί κανείς ότι πάρα πολλοί χρήστες χρησιμοποιούν – κακώς – το ίδιο συνθηματικό σε πολλές διαφορετικές εφαρμογές). Η επίθεση SQL injection παρουσιάζεται τα τελευταία χρόνια στην πρώτη θέση σε δύο από τις κορυφαίες λίστες απειλών διαδικτύου και επικινδυνότητας επιθέσεων, τη λίστα "Top 25 Most Dangerous Software Errors" των MITRE/SANS [05] και τη λίστα " Top 10 List " των OWASP [17].

Οι εφαρμογές χρειάζεται να χρησιμοποιούν κώδικα SQL για να διαχειρίζονται δεδομένα από και προς τον εξυπηρετητή και τη βάση δεδομένων τους. Το λογισμικό και οι εφαρμογές που αναπτύσσονται στη σημερινή εποχή είναι περισσότερο λογισμικό διαχείρισης δεδομένων παρά κάτι άλλο. Και η πιο απλή εφαρμογή σήμερα είναι πολύ πιθανό να υλοποιεί έστω και μια απλή βάση δεδομένων [05].

Η επίθεση αυτή εκμεταλλεύεται ευπάθειες που δημιουργούνται στα συστήματα όταν αυτά έχουν να διαχειριστούν δεδομένα από ανασφαλείς πηγές, με πιο ανασφαλή δεδομένα αυτά που προέρχονται κατευθείαν από το χρήστη. Συνεπώς, η επίθεση SQL injection αποτελεί υποσύνολο επιθέσεων που εκμεταλλεύονται ευπάθειες στα δεδομένα εισόδου των εφαρμογών. Η βασική ιδέα της επίθεσης είναι να "πειστεί" η εφαρμογή να εκτελέσει κακόβουλο κώδικα SQL, ο οποίος κανονικά δεν θα έπρεπε να εκτελεστεί [09]. Ο επιτιθέμενος προσπαθεί να εισάγει δεδομένα στο σύστημα με τέτοιο τρόπο έτσι ώστε να καταφέρει να αλλοιώσει τη δομή αυτού του κώδικα.

Όταν για παράδειγμα ο χρήστης μιας ιστοσελίδας εισάγει στην εφαρμογή το όνομα και το συνθηματικό του, η εφαρμογή συνθέτει με αυτά μια εντολή SQL η οποία θα εκτελεστεί στη βάση δεδομένων, για να αποφασιστεί αν ο χρήστης θα αποκτήσει πρόσβαση στο σύστημα ή όχι.



Σχήμα 5.1 – Παράδειγμα ορθής εισαγωγής δεδομένων

Στο παράδειγμα το σχήματος 5.1, το σύστημα θα συνθέσει – για παράδειγμα – την πιο κάτω εντολή SQL, την οποία και θα εκτελέσει στη βάση δεδομένων:

```
select *
from users
where username = 'Δημήτρης' and password = '1234';
```

Αν η εφαρμογή είναι ευάλωτη σε επίθεση SQL injection, τότε ο κακόβουλος χρήστης μπορεί να δώσει αλλοιωμένα δεδομένα στη θέση του ονόματος και συνθηματικού τα οποία να συνθέτουν μια κακόβουλη εντολή SQL. Θα μπορούσε για παράδειγμα κάποιος να αποκτήσει πρόσβαση στο σύστημα χωρίς να έχει στην κατοχή του κάποιο έγκυρο όνομα χρήστη ή συνθηματικό. Μπορεί απλά να τοποθετήσει το κατάλληλο κείμενο στο χώρο του username (Σχήμα 5.2).

Σχήμα 5.2 – Παράδειγμα επίθεσης SQL injection για μη εξουσιοδοτημένη πρόσβαση στο σύστημα

Σε αυτή την περίπτωση ο κώδικας SQL μετατρέπεται σε εντολή η οποία πάντα επιστρέφει σωστό αποτέλεσμα, λόγω του "or true":

```
select *
from users
where username = '' or true; -- ' and password = '';
```

Ο επιτιθέμενος, από τη στιγμή που έχει τη δυνατότητα να "κτίζει" δικό του κώδικα στην εντολή SQL του συστήματος, μπορεί να κάνει και άλλες ενέργειες στην ευάλωτη σελίδα, πέρα από την απόκτηση μη εξουσιοδοτημένης πρόσβασης [09]. Μπορεί, για παράδειγμα, να αλλοιώσει τον πίνακα users ή ακόμη και να τον διαγράψει. Για την εκτέλεση αυτών των επιθέσεων όμως, ο κακόβουλος χρήστης θα χρειαστεί επιπρόσθετα να μαντέψει το όνομα του πίνακα της υποκείμενης βάσης δεδομένων. Επίσης θα πρέπει η βάση δεδομένων μας να είναι λάθος παραμετροποιημένη και να επιτρέπει την εκτέλεση των εντολών DROP, INSERT και UPDATE από χρήστες του συστήματος. Ακολουθούν τα παραδείγματα:

- Διαγραφή του πίνακα users:

```
select *
from users
where username = 'x'; DROP TABLE users; --' and password = '';
```

Ο χρήστης χρησιμοποιεί το σύμβολο " ; " το οποίο σημαίνει το τέλος της εντολής SELECT. Στη συνέχεια προσθέτει την εντολή DROP, η οποία εκτελείται αυτόνομα από τη βάση δεδομένων. Τα σύμβολα " -- " χρησιμοποιούνται για να ακυρώσουν το υπόλοιπο της εντολής SELECT.

- Προσθήκη δεδομένων στον πίνακα:

```
select *
from users
where username = 'x';
  INSERT INTO users (username, password)
  VALUES ('user1', 'user1'); --' and password = '';
```

- Αλλαγή υφιστάμενων δεδομένων του πίνακα. Για παράδειγμα να αλλάξει το συνθηματικό του χρήστη "Δημήτρης" σε κάτι άλλο:

```
select *
from users
where username = 'x';
  UPDATE users
  SET password = myPass'
```

```
WHERE username = 'Δημήτρης' --' and password = '';
```

Πρέπει να σημειωθεί ότι το να μαντέψει ο επιτιθέμενος το όνομα κάποιου πίνακα της βάσης ή ονόματα γνωρισμάτων (πεδίων) αυτού δεν είναι απαραίτητα τόσο δύσκολο, εφόσον η διαδικτυακή εφαρμογή είναι πράγματι ευάλωτη σε επιθέσεις τύπου SQL injection. Και αυτό γιατί μπορεί, θέτοντας κατάλληλες εισόδους στη διαδικτυακή εφαρμογή βάσει του μαντέματός του (π.χ. μπορεί να εικάσει ότι θα υπάρχει ένας πίνακας με το όνομα "users" ή ένα πεδίο με το όνομα "email"), από το μήνυμα λάθους ή όχι που θα του επιστρέφει στην ιστοσελίδα ο διακομιστής διαδικτύου (web server) να αποφανθεί με βεβαιότητα αν η πρόβλεψή του ήταν σωστή. Δεδομένου ότι μπορεί να επιχειρήσει πολλές δοκιμές μέχρι να μαντέψει σωστά, γίνεται αντιληπτό ότι τέτοιες επιθέσεις είναι εξαιρετικά πιθανό να πραγματοποιηθούν (όπως άλλωστε καταδεικνύει και η πράξη).

Είναι λοιπόν απαραίτητο να θωρακίζουμε τις εφαρμογές μας εναντίων της επίθεσης SQL injection. Μερικά μέτρα τα οποία μπορούμε να λάβουμε είναι [05], [09], [17]:

- **Έλεγχος των δεδομένων** που διαχειρίζεται η εφαρμογή από μη ασφαλής πηγές. Ο έλεγχος που πρέπει να κάνουμε είναι στη μορφή των δεδομένων, το περιεχόμενο τους και το μέγεθος (για παράδειγμα, ειδικοί χαρακτήρες, όπως η απόστροφος ', πρέπει να αγνοούνται).

- **Δεν συνθέτουμε εντολές SQL** με δεδομένα από το χρήστη όπως αυτά εισάγονται. Στις περιπτώσεις που χρησιμοποιούμε δεδομένα από το χρήστη στις εντολές SQL θα πρέπει να τα χειριζόμαστε παραμετρικά, σε προ-κωδικοποιημένες εντολές SQL. Επίσης θα πρέπει να τα διορθώνουμε, μετατρέποντας "ύποπτους" χαρακτήρες όπως τα εισαγωγικά.

- **Σχεδιασμός και παραμετροποίηση της βάσης δεδομένων**. Φροντίζουμε η βάση δεδομένων μας να είναι σχεδιασμένη με τέτοιο τρόπο έτσι ώστε να επιτρέπει πρόσβαση από το σύστημα μόνο στις εντολές οι οποίες είναι απαραίτητες. Δεν πρέπει η βάση δεδομένων μας να επιτρέπει στο σύστημα να εκτελεί εντολές που έτσι κι αλλιώς (το σύστημα) δεν θα τις χρησιμοποιήσει. Περιορίζουμε έτσι τον επιτιθέμενο σε μικρότερο αριθμό πιθανών εντολών που θα μπορεί να εκτελέσει κακόβουλα.

- **Μηνύματα λάθους**. Τα μηνύματα που επιστρέφει η εφαρμογή μας σε περιπτώσεις λάθους δεν πρέπει να δίνουν λεπτομέρειες σχετικές με τη βάση δεδομένων, ονόματα πινάκων και πεδίων ή ακόμη και ονόματα παραμέτρων και μεθόδων του κώδικα μας. Τέτοιες πληροφορίες δίνουν στον κακόβουλο χρήστη τον τρόπο να εκτελέσει στοχοποιημένες επιθέσεις σε συγκεκριμένα καίρια σημεία της εφαρμογής (όπως για παράδειγμα ο πίνακας με τους χρήστες και τα συνθηματικά τους).

5.2 Επιθέσεις Cross Site Scripting (XSS)

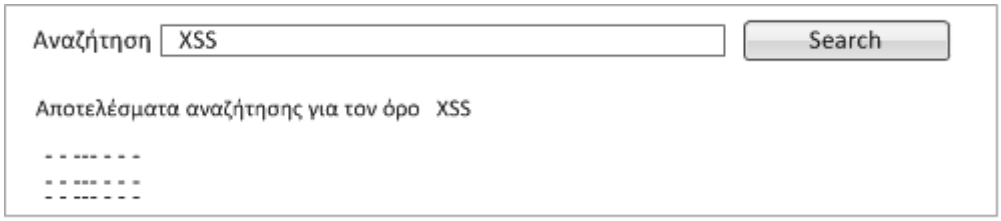
Η επίθεση αυτή, αποτελεί μια από τις πιο διαδεδομένες επιθέσεις στο διαδίκτυο. Οι λίστες απειλών διαδικτύου και επικινδυνότητας επιθέσεων ([05], [17]) την τοποθετούν στην 3η και 4η θέση το 2013 και 2011 αντίστοιχα. Μέσω αυτής της επίθεσης ο φυλλομετρητής του αθώου χρήστη εκτελεί κακόβουλο κώδικα με μια απλή επίσκεψη του σε μια κανονική κατά τα άλλα ιστοσελίδα. Ο επιτιθέμενος χρησιμοποιεί κακόβουλα τμήματα κώδικα (scripts) που εκτελούνται στο φυλλομετρητή του χρήστη (HTML και JavaScript). Η επηρεαζόμενη ιστοσελίδα στέλνει τον κακόβουλο κώδικα στους χρήστες χωρίς να το γνωρίζουν και οι φυλλομετρητές τους τον εκτελούν. Ο κώδικας θεωρείται ασφαλής από το φυλλομετρητή αφού προήλθε από σελίδα η οποία είναι έμπιστη [34].

Πολλές φορές ο επιτιθέμενος "κατασκευάζει" διευθύνσεις URL οι οποίες ενώ δείχνουν στη σωστή και έμπιστη σελίδα, περιέχουν επιπρόσθετο κακόβουλο κώδικα. Το μόνο που έχει να κάνει ο επιτιθέμενος είναι να διανείμει τη "μολυσμένη" διεύθυνση URL στους χρήστες (π.χ. μέσω παραπλανητικών μηνυμάτων ηλεκτρονικού ταχυδρομείου, ή σε forums/κοινωνικά δίκτυα κτλ.) οι οποίοι εκτελούν έτσι τον κώδικα από το φυλλομετρητή τους κατά τη μετάβαση στη σελίδα.

Κάποιος κακόβουλος χρήστης μπορεί για παράδειγμα να προσέξει ότι όταν αναζητήσει σε διαδικτυακή εφαρμογή κάποιο όρο, τότε η εφαρμογή τοποθετεί τον όρο αυτό στη διεύθυνση url, και εκτελώντας τη διεύθυνση αυτή η σελίδα δείχνει το κείμενο "Αποτελέσματα αναζήτησης για τον όρο …" μαζί με τα αποτελέσματα. Έτσι, όταν αναζητήσει κάποιος τον όρο "XSS" τότε η διεύθυνση URL γίνεται:

```
http://www.example.cy?q=XSS
```

ενώ τα αποτελέσματα παρουσιάζονται στη μορφή του σχήματος 5.3.

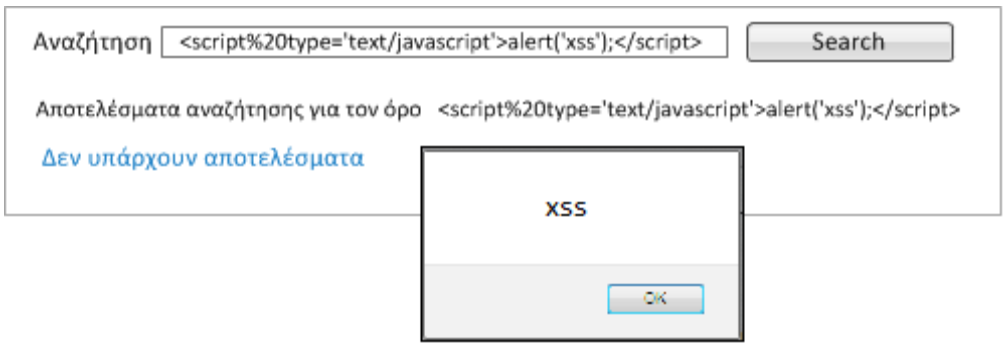


Σχήμα 5.3 – Αναζήτηση του όρου XSS

Όταν όμως ο κακόβουλος χρήστης δημιουργήσει κατευθείαν τη διεύθυνση URL με περιεχόμενο αναζήτησης κάποιο κώδικα JavaScript,

```
http://www.example.cy?q=<script%20type='text/javascript'>alert('xss');</script>
```

και τη διαδώσει στο διαδίκτυο, τότε ο χρήστης – θύμα, θα "αναγκάσει" το φυλλομετρητή του να εκτελέσει και τον κώδικα JavaScript, αφού είναι ρυθμισμένος να εκτελεί κώδικα JavaScript όταν τον λαμβάνει (Σχήμα 5.4). Παρόλο που σε αυτή την περίπτωση θα λάβει απλά το μήνυμα "xss" σε αναδυόμενο παράθυρο (λόγω της εντολής "alert"), ο κακόβουλος κώδικας θα μπορούσε να κάνει άλλες ενέργειες, οι οποίες μάλιστα δεν γίνονται αντιληπτές από το χρήστη (π.χ. να κλέψει authentication cookies και να τα στείλει στον επιτιθέμενο). Σημειώνουμε ότι η σελίδα που επισκέφτηκε ο χρήστης – θύμα είναι η κανονική (αθώα) σελίδα την οποία ο χρήστης – θύμα εμπιστεύεται, και όχι κάποια πλαστή. Ο κακόβουλος κώδικα κρύβεται σε τμήμα της διεύθυνσης URL.



Σχήμα 5.4 – Επίθεση XSS.

Είναι σημαντικό να σημειωθεί ότι τα συμβατικά μέτρα ασφαλείας (firewalls, intrusion detection systems, anti-virus) δεν μπορούν να προστατεύσουν το χρήστη από αυτή την επίθεση [06]. Για την προστασία από αυτή την επίθεση δεν υπάρχουν μέτρα τα οποία μπορεί να λάβει ο χρήστης μιας εφαρμογής για να προστατευτεί πλήρως. Το μόνο μέτρο το οποίο συστήνεται για τους χρήστες είναι να απενεργοποιεί από το φυλλομετρητή του τη χρήση scripts [04]. Αυτό το μέτρο όμως θεωρείται ακραίο, καθώς με αυτό τον τρόπο απενεργοποιούνται και ένα σωρό άλλες χρήσεις των scripts οι οποίες είναι απαραίτητες για την περιήγηση μας στο διαδίκτυο.

Οι επιθέσεις Cross Site Scripting μπορούν να προληφθούν σε επίπεδο εφαρμογής. Για να προστατεύσουμε τις εφαρμογές μας από επιθέσεις Cross-Site Scripting, μπορούμε να λάβουμε τα πιο κάτω μέτρα:

- Δεν συνθέτουμε κώδικα HTML, PHP κλπ μαζί με δεδομένα από μη ασφαλείς πηγές. Η καρδιά του προβλήματος βρίσκεται σε αυτό το σημείο. Ο κακόβουλος κώδικας εισέρχεται στην εφαρμογή μας συνθέτοντας κώδικα της εφαρμογής μαζί με κείμενο το οποίο λάβαμε από κάποιο χρήστη ή από άλλη μη ασφαλή πηγή.

- Διορθώνουμε δεδομένα που προέρχονται από μη ασφαλείς πηγές μετατρέποντας χαρακτήρες που χρησιμοποιούνται και από γλώσσες scripting στα αντίστοιχα τους hex [18]. Για παράδειγμα:

```
& --> &amp;
< --> &lt;
> --> &gt;
" --> &quot;
```

- Καθαρίζουμε τον κώδικα μας με τη χρήση εργαλείων που βρίσκουν ευπάθειες που μπορεί να προκαλέσουν επιθέσεις XSS. Ένα καλό εργαλείο είναι το HtmlSanitizer [13], [18].

Κεφάλαιο 6
Εικονικό Εργαστήριο Κρυπτογραφίας και Ασφάλειας Πληροφοριών

Στο παρόν κεφάλαιο παρουσιάζεται το κύριο μέρος της παρούσας διατριβής, μέσω αναλυτικής περιγραφής των εφαρμογών που αναπτύχθηκαν στο πλαίσιο της δημιουργίας ενός εικονικού εργαστηρίου κρυπτογραφίας και ασφάλειας πληροφοριών. Ουσιαστικά, η δημιουργία του εν λόγω εργαστηρίου αποσκοπεί στην εξοικείωση και πρακτική εξάσκηση των φοιτητών ως προς όλα τα ζητήματα ασφάλειας που περιγράφηκαν νωρίτερα. Στο κεφάλαιο αυτό παρουσιάζεται, μεταξύ άλλων, το εικονικό εργαστήριο, με περιγραφή τόσο των τεχνολογιών που αξιοποιήθηκαν για τη δημιουργία του όσο και του τρόπου χρήσης αυτού.

6.1 Εκπαιδευτικό λογισμικό

Εκπαιδευτικό λογισμικό θεωρούμε το λογισμικό το οποίο βοηθά στη διαδικασία της μάθησης. Με τη χρήση του (εκπαιδευτικού λογισμικού) προσπαθούμε να διδάξουμε ένα γνωστικό αντικείμενο, με τρόπο που πολλές φορές φαίνεται να αντικαθιστά την τάξη και τον

εκπαιδευτικό. Με το εκπαιδευτικό λογισμικό γίνεται πιο εύκολη η εξατομικευμένη μάθηση, αφού το λογισμικό προσαρμόζεται στις ανάγκες του κάθε εκπαιδευόμενου [38], [41].

Κατηγορίες εκπαιδευτικού λογισμικού

Το εκπαιδευτικό λογισμικό κατηγοριοποιείται και ταξινομείται ανάλογα με τη χρήση του σε πέντε κύριες κατηγορίες ([20],[40],[41]):

Λογισμικό εξάσκησης (Drill & Practice). Εκπαιδευτικό λογισμικό το οποίο προσφέρει τη δυνατότητα στο φοιτητή να εξασκηθεί σε κάποιο θέμα το οποίο έχει ήδη διδαχθεί. Το λογισμικό παρουσιάζει ασκήσεις και προβλήματα τα οποία ο εκπαιδευόμενος καλείται να λύσει. Οι ασκήσεις συνήθως βρίσκονται αποθηκευμένες σε βάση δεδομένων και παρουσιάζονται στο εκπαιδευόμενο με τυχαία σειρά, με σειρά βαθμού δυσκολίας, κλπ. Με τη χρήση του λογισμικού εξάσκησης, ο καθηγητής και ο εκπαιδευόμενος μπορούν να έχουν άμεση ανατροφοδότηση, ενώ υπάρχει και η δυνατότητα καταγραφής αποτελεσμάτων και προόδου.

Λογισμικό Παρουσίασης (Tutorial). Συνδυάζει πολυμέσα (κείμενο, εικόνα, ήχος, βίντεο) με σκοπό να παρουσιαστούν στον εκπαιδευόμενο νέες έννοιες και αδίδακτες πληροφορίες. Μοιάζει πολύ με κατ' ιδίαν διδασκαλία. Σε συγκεκριμένα σημεία, παρέχονται στον εκπαιδευόμενο πηγές και σύνδεσμοι, έτσι ώστε να μπορεί, αν θέλει, να διερευνήσει σε μεγαλύτερο βάθος. Κατά τη διάρκεια της παρουσίασης των νέων πληροφοριών το λογισμικό μπορεί να θέσει ερωτήσεις προς το χρήστη με σκοπό να διαπιστωθεί ο βαθμός κατανόησης. Το λογισμικό επαναλαμβάνει και επεξηγεί περισσότερο τα σημεία που χρειάζεται.

Εκπαιδευτικό παιχνίδι (Educational game). Οι εκπαιδευτικοί στόχοι προωθούνται αξιοποιώντας το περιβάλλον ενός παιχνιδιού. Ο εκπαιδευόμενος κινητοποιείται μέσω του ενθουσιασμού που δημιουργεί το παιγνιώδες περιβάλλον.

Προσομοίωσης (Simulation). Στα προγράμματα προσομοίωσης, ο εκπαιδευόμενος έχει την ευκαιρία να πειραματιστεί και να εξασκηθεί, σε ένα τεχνητό περιβάλλον. Χρησιμοποιείται σε περιπτώσεις που η εξάσκηση σε πραγματικό περιβάλλον είναι επικίνδυνη ή πολύ ακριβή. Ο εκπαιδευόμενος μπορεί να αλλάζει τις παραμέτρους στο περιβάλλον αυτό και να παρατηρεί και καταγράφει τα αποτελέσματα.

Επίλυσης προβλημάτων (Problem solving). Το λογισμικό αυτό παρουσιάζει στον εκπαιδευόμενο κάποιο πρόβλημα το οποίο αναμένεται να λύσει χρησιμοποιώντας γνώσεις που ήδη κατέχει. Τα προβλήματα τα οποία παρουσιάζονται στοχεύουν στην ανάπτυξη ικανοτήτων επίλυσης προβλημάτων, και όχι στην μετάδοση γνώσης.

6.2 Η εφαρμογή ως εκπαιδευτικό λογισμικό

Η εφαρμογή που αναπτύχθηκε στην παρούσα διατριβή ανήκει κυρίως στην κατηγορία **Λογισμικό Προσομοίωσης**. Προσφέρεται στο φοιτητή-χρήστη ένα περιβάλλον για να πειραματιστεί στις επιθέσεις ασφάλειας που παρουσιάζονται και να εξοικειωθεί με αυτές.

Στην εφαρμογή μας παρουσιάζονται επίσης στοιχεία **Λογισμικού Παρουσίασης**, αφού παρουσιάζει και επεξηγεί στο χρήστη τον τρόπο με τον οποίο πραγματοποιούνται οι επιθέσεις, τις ευπάθειες που εκμεταλλεύονται οι επιτιθέμενοι και τα μέτρα με τα οποία μπορεί να προστατευτεί κάποιος από αυτές.

6.3 Τι είναι ένα εργαστήριο κρυπτογραφίας και ασφάλειας πληροφοριών

Η ασφάλεια συστημάτων παρουσιάζει δυσκολίες ως προς την εξάσκηση των φοιτητών στα θέματα αυτά. Δεν μπορεί κάποιος να αρχίσει να δοκιμάζει τις γνώσεις του πάνω σε τυχαίες "ζωντανές" σελίδες που βρίσκονται στο διαδίκτυο. Οι φοιτητές χρειάζονται κάποιο χώρο όπου να μπορούν νόμιμα να πειραματίζονται σε θέματα ασφάλειας και να μπορούν να εξασκούνται σε ευπάθειες συστημάτων [03].

Εργαστήριο κρυπτογραφίας και ασφάλειας πληροφοριών ονομάζουμε μια πλατφόρμα εφαρμογών οι οποίες ομαδοποιήθηκαν και συνδέθηκαν σε μια κεντρική ιστοσελίδα. Οι εφαρμογές αυτές αφορούν θέματα ασφάλειας συστημάτων και λειτουργούν σε ένα ελεγχόμενο χώρο στον οποίο ο χρήστης μπορεί να πειραματισθεί και εξασκηθεί νόμιμα σε επιθέσεις και απειλές κατά συστημάτων. Αποτελεί εκπαιδευτικό υλικό, επικαιροποιημένο ως προς τις σύγχρονες εξελίξεις του χώρου, το οποίο αποτελεί εξαιρετικό εργαλείο για την πρακτική τους εξάσκηση.

6.4 Ευπάθειες και άλλα θέματα που περιλαμβάνονται στην εφαρμογή

Για την επιλογή των θεμάτων που περιλαμβάνονται στην εφαρμογή μας, λάβαμε υπόψη τους εξής παράγοντες:

- **Βαθμός δυσκολίας**. Το θέμα που θα επιλεγεί θα απευθύνεται σε φοιτητές οι οποίοι διδάσκονται πρώτη φορά θέματα ασφάλειας υπολογιστών. Οι ευπάθειες και επιθέσεις οι οποίες θα συμπεριληφθούν θα πρέπει να μπορούν να γίνουν εύκολα κατανοητές, να αφορούν ευπάθειες σε σημεία τα οποία είναι γνωστά (όχι κενά ασφαλείας σε ειδικευμένα κομμάτια λογισμικού που δεν γνωρίζει ο μέσος φοιτητής θεμάτων πληροφορικής) και να μπορούμε να τα επιδείξουμε με καθαρό και κατανοητό τρόπο.

- **Χρησιμότητα και σχετικότητα** με τη κοινότητα και βιομηχανία πληροφορικής. Οι επιθέσεις οι οποίες θα συμπεριλαμβάνονται θα πρέπει να είναι συνήθεις στη σημερινή πραγματικότητα. Θα πρέπει να συμπεριληφθούν επιθέσεις οι οποίες πραγματικά προκαλούν σημαντικά προβλήματα στο χώρο του διαδικτύου και ο χρήστης να μάθει για ευπάθειες οι οποίες θα είναι σχετικές με τη σημερινή κατάσταση της βιομηχανίας. Θα πρέπει να είναι οι βασικές επιθέσεις τις οποίες ο σημερινός επαγγελματίας ασφάλειας είναι απόλυτα αναγκαίο να γνωρίζει καλά.

- **Άλλες πηγές και εργαλεία**. Παρόλο που η εφαρμογή μας δεν είναι μοναδική στο είδος της, θα συμπεριλάβουμε θέματα τα οποία δεν βρίσκουμε στο βαθμό που θα περιμέναμε και στη μορφή που θα θέλαμε στα άλλα εργαλεία.

Μετά από μελέτη των πιο πάνω παραγόντων και κριτηρίων, καταλήξαμε ότι θα συμπεριληφθούν στην εφαρμογή τα πιο κάτω θέματα:

SQL Injection

Η επίθεση SQL Injection αποτελεί την πιο διαδεδομένη μορφή επίθεσης σε εφαρμογές και ιστοσελίδες τα τελευταία χρόνια [05], [17]. Αποτελεί την κύρια απειλή σε ιστοσελίδες και όλοι οι επαγγελματίες στο χώρο της πληροφορικής θα πρέπει να είναι εξοικειωμένοι με την επίθεση, τις ευπάθειες και τα μέτρα πρόληψης τα οποία πρέπει να λαμβάνονται.

Εκτός από τη σημαντικότητα της επίθεσης αυτής, διαπιστώθηκε στο πλαίσιο της διατριβής ότι παρόλο που η επίθεση επεξήγεται και προσομοιώνεται και σε άλλα εργαλεία παρόμοιου χαρακτήρα με το δικό μας (πχ WebGoat [03]), απευθύνεται σχεδόν πάντα σε άτομα που είναι ήδη επαγγελματίες στο χώρο, και όχι σε φοιτητές οι οποίοι δεν έχουν διδακτεί ακόμη τις σχετικές έννοιες.

Σκοπός μας είναι το θέμα SQL injection το οποίο θα συμπεριληφθεί στην εφαρμογή να περιγράφει την επίθεση στην απλή της μορφή και ο φοιτητής να μπορεί να πειραματιστεί σε αυτήν. Ο φοιτητής θα έχει άμεση ανατροφοδότηση από το σύστημα για θέματα όπως "η επίθεση υπήρξε επιτυχείς επειδή…" ή "η επίθεση δεν ήταν επιτυχείς επειδή…".

Επίθεση λεξικού και σωστή χρήση συνθηματικών

Θεωρήσαμε αναγκαίο να συμπεριλάβουμε ένα θέμα το οποίο να ασχολείται με επιθέσεις σχετικές με τη σωστή χρήση συνθηματικών. Οι λόγοι στους οποίους στηρίχτηκε η επιλογή αυτή είναι:

- Οι επιθέσεις σχετικές με τη λανθασμένη χρήση συνθηματικών και ευπάθειες σε κώδικα πιστοποίησης και αυθεντικοποίησης είναι, ακόμα και σήμερα, πολύ συνηθισμένες και εμφανίζονται σε διάφορες λίστες που καταγράφουν επικίνδυνες ευπάθειες και επιθέσεις ([05], [17]).

- Η διδασκαλία της επίθεσης αυτής αποτελεί φυσική συνέχεια αυτής της SQL injection. Στις περιπτώσεις που ο επιτιθέμενος ενδιαφέρεται να αποκτήσει πρόσβαση σε μια ιστοσελίδα ή ένα σύστημα, και όταν αποτύχει να εξασφαλίσει συνθηματικά μιας με επίθεση SQL injection, συνήθως καταφεύγει σε επιθέσεις λεξικού.

Κρυπτογραφία και επίθεση γνωστού μηνύματος

Θέματα κρυπτογραφίας, κρυπτογραφικές ευπάθειες και επιθέσεις σε κρυπτογραφημένα μηνύματα δεν καλύπτονται σε παρόμοιου τύπου εφαρμογές. Η κρυπτογραφία όμως είναι από τα θέματα στα οποία ο φοιτητής χρειάζεται επίδειξη και πρακτική εξάσκηση για να κατανοήσει πλήρως. Η αξία της πρακτικής εξάσκησης είναι ακόμη μεγαλύτερη όταν ο φοιτητής καλείται να κατανοήσει θέματα ευπαθειών και επιθέσεων σε κρυπτοκείμενα. Η επίδειξη επίθεσης γνωστού

μηνύματος προσφέρεται για την εκμάθηση και εξάσκηση σε κρυπτογραφικούς αλγόριθμους ροής (συμμετρική κρυπτογράφηση).

WebGoat

Στην εφαρμογή μας θα συμπεριλάβουμε και ένα θέμα για το εργαλείο WebGoat [03]. Το WebGoat είναι η πιο γνωστή και αξιόλογη εφαρμογή για εκμάθηση προχωρημένων εννοιών ασφάλειας, ευπαθειών και επιθέσεων. Περιλαμβάνουμε το θέμα αυτό για να γνωρίσει ο φοιτητής μια εφαρμογή προχωρημένου επιπέδου, την οποία όλοι οι επαγγελματίες στα θέματα ασφάλειας πρέπει να γνωρίζουν.

6.5 Λειτουργικές απαιτήσεις και σχεδιασμός

Η εφαρμογή που θα αναπτυχθεί θα έχει τη μορφή ιστοσελίδας, η οποία θα περιλαμβάνει μια υπο-σελίδα για κάθε θέμα (ευπάθεια, επίθεση, κα). Η υπο-σελίδα του κάθε θέματος αναπτύσσεται αυτόνομα και ξεχωριστά από τις σελίδες των άλλων θεμάτων. Το μοντέλο αυτό μας επιτρέπει να προσθέτουμε και να αναβαθμίζουμε τη σελίδα μας με νέα θέματα πολύ εύκολα, με σχεδόν κανένα περιορισμό σχετικό την υφιστάμενη εφαρμογή, ακόμα και μετά την υλοποίηση και εγκατάσταση.

Οι μόνες εξαρτήσεις που θέτουμε κατά το σχεδιασμό και κατασκευή των υπο-σελίδων της εφαρμογής αφορούν τη χρήση κοινής βάσης δεδομένων και τη χρήση κοινού αρχείου css.

Επιλέγουμε να χρησιμοποιήσουμε μια **κοινή βάση δεδομένων** για όλες τις σελίδες, αντί μίας ξεχωριστής για κάθε θέμα, για τους ακόλουθους λόγους:

- Με τη χρήση μίας μόνο βάσης δεδομένων περιορίζουμε τις ελάχιστες απαιτήσεις και προδιαγραφές που ζητούμε για τον web server μας.

- Απλοποιείται η διαδικασία εγκατάστασης της εφαρμογής. Μπορούμε πολύ πιο εύκολα να προσθέσουμε και να εγκαταστήσουμε σελίδες όταν χρησιμοποιούν την υφιστάμενη κοινή βάση δεδομένων, αντί να χρειάζεται κάθε φορά να εγκαθιστούμε νέα βάση δεδομένων.

Επιλέγουμε να χρησιμοποιήσουμε **κοινό αρχείο css** για σκοπούς εμφάνισης και παρουσίασης της σελίδας. Παρόλο που οι σελίδες για κάθε θέμα θα αναπτύσσονται ξεχωριστά η μία από την άλλη, η εφαρμογή παραμένει ενιαία και πρέπει όλες οι υποσελίδες της να παρουσιάζονται με παρόμοιο τρόπο (κοινά χρώματα, γραμματοσειρά, κλπ).

Στη συνέχεια θα αναλυθούν οι λειτουργικές απαιτήσεις και σχεδιαστικές λεπτομέρειες για κάθε ένα από τα θέματα θα περιλαμβάνει η εφαρμογή.

6.5.1 SQL Injection

Αυτό το θέμα απευθύνεται σε φοιτητές που δεν γνωρίζουν τι είναι η επίθεση SQL injection. Ακολουθώντας αυτό το θέμα, ο φοιτητής θα πρέπει να μπορεί να κατανοήσει τι είναι η επίθεση SQL injection, πώς πραγματοποιείται και ποια μέτρα προστασίας πρέπει να λαμβάνονται.

Η σελίδα θα περιλαμβάνει τα πιο κάτω τμήματα:

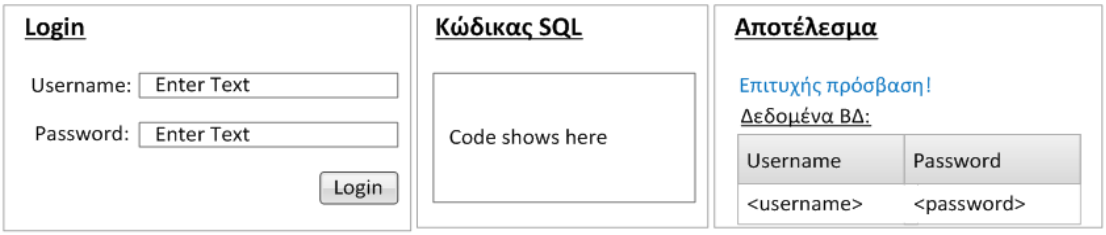
- ***Περιγραφή της επίθεσης***. Πώς πραγματοποιείται μία επίθεση SQL injection με παραδείγματα. Στα παραδείγματα θα πρέπει να υπάρχει και ανάλυση του κώδικα της επίθεσης.

- ***Επίδειξη επίθεσης***. Ο φοιτητής θα πρέπει να έχει τη δυνατότητα να επιχειρήσει την επίθεση μόνος του, σε πραγματικό περιβάλλον. Θα υλοποιηθεί μια απλή εφαρμογή διαδικτυακής σύνδεσης (login), η οποία θα είναι συνδεδεμένη με βάση δεδομένων και η οποία θα είναι ευάλωτη σε επιθέσεις SQL injection. Ο χρήστης θα μπορεί να βάζει δεδομένα στα πεδία "όνομα χρήστη" (username) και "συνθηματικό" (password) της εφαρμογής και η εφαρμογή θα εμφανίζει στο χρήστη:

  α) τον κώδικα SQL που δημιουργείται και θα τρέξει η εφαρμογή, και

  β) το αποτέλεσμα που επιστρέφει η βάση δεδομένων. Το αποτέλεσμα θα είναι ανάλογο με το δεδομένο κώδικα SQL που χρησιμοποιεί η εφαρμογή.

  Έτσι ο χρήστης θα έχει τη δυνατότητα να πειραματιστεί όσες φορές θέλει και με ό,τι δεδομένα εισόδου θέλει, για να προσπαθήσει να αποσπάσει συνθηματικά χρηστών από τη βάση δεδομένων.



Σχήμα 6.1: SQL Injection – Διεπαφή Login

- ***Προστασία***. Σε αυτό το τμήμα της εφαρμογής θα πρέπει να υπάρχει κατάλογος με τα μέτρα που μπορούμε να παίρνουμε για να προστατεύσουμε τις εφαρμογές μας από επιθέσεις SQL injection.

- ***Βάση δεδομένων***. Σε αυτό το θέμα, θα χρειαστεί να δημιουργήσουμε ένα πίνακα **Users**, ο οποίος θα χρησιμοποιείται από την εφαρμογή login (***Επίδειξη επίθεσης***). Ο πίνακας αυτός θα είναι ο στόχος του χρήστη της εφαρμογής.

Πίνακας:	**Users**
Πεδία:	user ID (integer) email (string) username(string) password (string) role (string)

Σχήμα 6.2: SQL Injection – Ο πίνακας users

Για τους σκοπούς της επίδειξης της επίθεσης, ο πίνακας θα περιλαμβάνει μικρό αριθμό εγγραφών.

6.5.2 Επίθεση λεξικού και σωστή χρήση συνθηματικών

Σε αυτό το θέμα ο φοιτητής θα μάθει τι είναι η επίθεση λεξικού (Dictionary attack) και ποιοι είναι οι κίνδυνοι της επίθεσης αυτής, ακόμη και όταν τα συνθηματικά είναι κωδικοποιημένα στη βάση δεδομένων. Επίσης, ο φοιτητής θα είναι σε θέση να γνωρίζει τα μέτρα προστασίας από αυτή την επίθεση.

Μέσα από το θέμα αυτό, ο φοιτητής θα έρθει σε επαφή με καλές πρακτικές για τη σωστή χρήση των συνθηματικών.

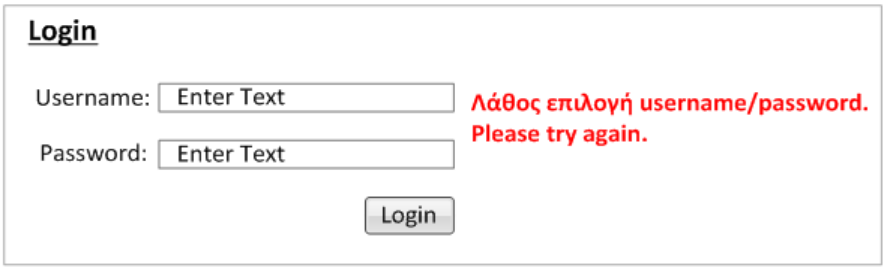
Η σελίδα θα δίνει την ευκαιρία στο φοιτητή να δοκιμάσει να εκτελέσει τη δική του επίθεση, μέσα από κάποιο σενάριο που θα δοθεί. Θα περιλαμβάνει τα εξής τμήματα:

- ***Περιγραφή της επίθεσης***. Εδώ θα επεξηγείται η επίθεση. Το κείμενο να απευθύνεται σε άτομα τα οποία δεν έχουν προηγούμενη γνώση του αντικειμένου.

- ***Επίδειξη επίθεσης***. Ο φοιτητής έχει τη δυνατότητα να εκτελέσει επίθεση λεξικού στη βάση δεδομένων της εφαρμογής. Αυτό θα γίνει μέσα από ένα σενάριο το οποίο θα καθοδηγεί το φοιτητή βήμα προς βήμα. Μέσα από το σενάριο, θα δοθεί στο φοιτητή έτοιμος ο πίνακας users της βάσης δεδομένων, στον οποίο όμως τα συνθηματικά των χρηστών είναι κατακερματισμένα με τον αλγόριθμο SHA256.

  Το σενάριο ζητά από το χρήστη της εφαρμογής να συντάξει κατάλογο με πιθανά συνθηματικά (λεξικό) τα οποία και θα κωδικοποιήσει ο ίδιος με τον αλγόριθμο SHA256 – δηλαδή θα υπολογίσει την κατακερματισμένη τους τιμή ή, αλλιώς, το αποτύπωμά τους. Αν κάποιο από αυτά δώσει το ίδιο αποτύπωμα με κάποιο από τα κωδικοποιημένα συνθηματικά της βάσης δεδομένων, τότε αυτό το συνθηματικό είναι σωστό και θα επιτρέψει μη εξουσιοδοτημένη πρόσβαση.

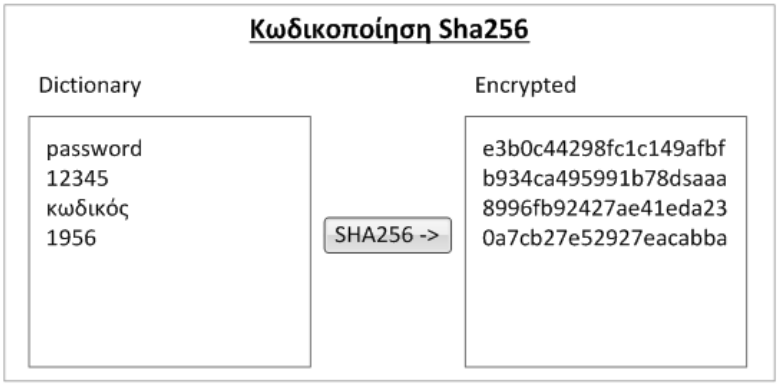
  Η σελίδα θα δίνει μια απλή εφαρμογή login, στην οποία ο φοιτητής θα μπορεί να χρησιμοποιήσει τα συνθηματικά που βρήκε και να επαληθεύσει την ορθότητα τους.



Σχήμα 6.3: Επίθεση λεξικού – Διεπαφή Login

Για τη σύνταξη του καταλόγου πιθανών συνθηματικών, δίνονται στο φοιτητή πληροφορίες για κάθε χρήστη του συστήματος (Ημερ. γέννησης, αρ. τηλεφώνου, κλπ). Αυτό γίνεται ακριβώς γιατί, ακόμα και σήμερα, διαπιστώνεται στην πράξη ότι πολλοί χρήστες επιλέγουν συνθηματικά ευκολομνημόνευτα για αυτούς, βάσει τέτοιου τύπου προσωπικών τους πληροφοριών [30].

Επίσης, χρειάζεται να υλοποιηθεί εφαρμογή κωδικοποίησης (συνάρτησης κατακερματισμού) SHA256. Θα δέχεται τον κατάλογο (dictionary) σε απλό κείμενο και θα κωδικοποιεί τα στοιχεία του με τον αλγόριθμο SHA256.



Σχήμα 6.4: Επίθεση λεξικού – Κατακερματισμός με τη συνάρτηση SHA256

- ***Προστασία***. Σε αυτό το τμήμα της εφαρμογής, θα επεξηγούνται τα μέτρα που πρέπει να λαμβάνονται και οι καλές πρακτικές που πρέπει να τηρούνται έτσι ώστε να προστατεύουμε τις εφαρμογές μας από επιθέσεις αυτού του τύπου.

- ***Βάση δεδομένων***. Χρειάζεται να δημιουργήσουμε έναν πίνακα χρηστών, με τα συνθηματικά τους όμως να είναι κατακερματισμένα (κωδικοποιημένα) με τη συνάρτηση SHA256. Ο πίνακας θα χρησιμοποιηθεί από την εφαρμογή login (Επίδειξη επίθεσης).

Πίνακας:	**UsersSHA**
Πεδία:	User ID (integer) email (string) username (string) passwordSHA (string) role (string)

Σχήμα 6.5: Επίθεση λεξικού – Πίνακας usersSHA

Στον πίνακα θα τοποθετήσουμε τα στοιχεία που δίνονται στο χρήστη στην αρχή της επίδειξης επίθεσης.

6.5.3 Κρυπτογραφία – Επίθεση γνωστού μηνύματος

Σε αυτή τη σελίδα θα καλύψουμε την ***επίθεση γνωστού μηνύματος***, σε κρυπτογραφημένο κείμενο αλγόριθμου ροής (Συμμετρική κρυπτογράφηση).

Στόχος είναι να γίνει επίδειξη της σημασίας της γραμμικής πολυπλοκότητας της κλειδοροής/ακολουθίας ως προς μέτρο αποτίμησης της προβλεψιμότητας της, και πώς η χρήση ακολουθιών με μεγάλη γραμμική πολυπλοκότητα είναι απαραίτητη για να προστατευτεί το κρυπτοκείμενο μας από τις επιθέσεις γνωστού μηνύματος.

Η εφαρμογή θα δίνει τη δυνατότητα στο χρήστη να κρυπτογραφήσει ένα μήνυμα, με επιλογή κλειδοροής μεγάλης ή μικρής γραμμικής πολυπλοκότητας. Στη συνέχεια η εφαρμογή θα επιχειρεί κρυπτανάλυση με επίθεση γνωστού κειμένου (με τη χρήση του αλγόριθμου Berlekamp-Massey) και θα επιδεικνύει το αποτέλεσμα. Ο φοιτητής θα μπορεί να πειραματιστεί με διαφορετικά μεγέθη γραμμικής πολυπλοκότητας αλλά και διαφορετικά μεγέθη γνωστού κειμένου. Κλειδοροές με μεγάλη γραμμική πολυπλοκότητα θα δίνουν κρυπτοκείμενο που χρειάζεται πολύ μεγάλο γνωστό κείμενο για να αποκρυπτογραφηθεί (μέχρι και ολόκληρο το κείμενο, καθιστώντας ουσιαστικά αδύνατη την αποτελεσματική εφαρμογή της επίθεσης). Ενώ, αντίθετα, κλειδοροές με μικρή γραμμική πολυπλοκότητα θα παράγουν κρυπτοκείμενο που χρειάζεται μικρό γνωστό κείμενο για να αποκρυπτογραφηθεί.

Πιο συγκεκριμένα, η σελίδα αυτή περιλαμβάνει:

- ***Θεωρητικό υπόβαθρο*** για αλγόριθμους ροής. Σύντομα και περιεκτικά θα επεξηγήσουμε πώς λειτουργεί η κρυπτογράφηση με αλγόριθμους ροής. Σκοπός του τμήματος αυτού είναι να θέσει το θεωρητικό υπόβαθρο για την περιγραφή της επίθεσης.

- ***Περιγραφή επίθεσης***. Σε αυτό το τμήμα περιγράφουμε την επίθεση γνωστού κειμένου. Ο φοιτητής θα κατανοήσει τον τρόπο με τον οποίο γίνεται η επίθεση αλλά και να γνωρίσει τον αλγόριθμο με τον οποίο πραγματοποιείται (η επίθεση).

- ***Επίδειξη επίθεσης***. Ο φοιτητής θα μπορεί να πειραματιστεί και να πραγματοποιήσει επιθέσεις γνωστού κειμένου.

  Η επίδειξη θα είναι χωρισμένη σε τρία μέρη:

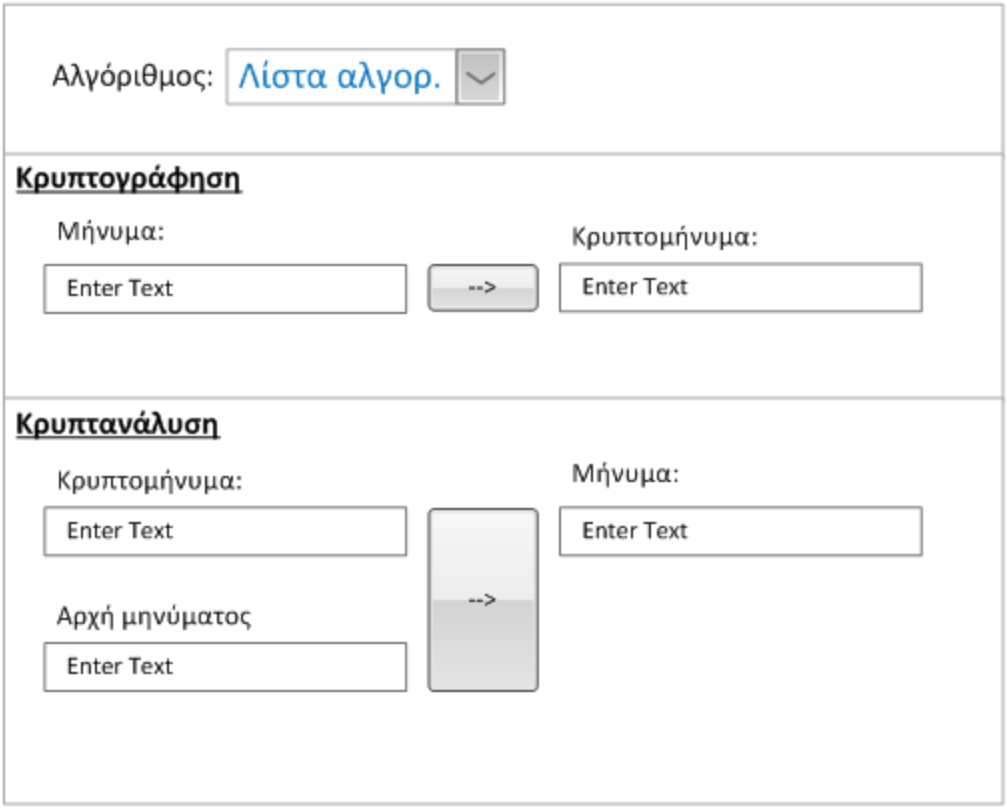
  - Επιλογή γεννήτριας κλειδοροής. Ο φοιτητής επιλέγει τον αλγόριθμο με τον οποίο θα παραχθεί η κλειδοροή. Θα παρέχονται επιλογές με διάφορα μεγέθη γραμμικής πολυπλοκότητας:

- Τυχαία κλειδοροή (υψηλή γραμμική πολυπλοκότητα).
- Πρωταρχικός LFSR με 10 <= n <= 50
- Πρωταρχικός LFSR με 51 <= n <= 100
- Πρωταρχικός LFSR με 101 <= n <= 200

- Κρυπτογράφηση. Ο φοιτητής θα δώσει ένα μήνυμα προς κρυπτογράφηση, και η εφαρμογή θα το κρυπτογραφήσει με αλγόριθμο ροής. Η κλειδοροή που θα χρησιμοποιηθεί κατά την κρυπτογράφηση θα παραχθεί από την εφαρμογή με βάση αλγόριθμο που επέλεξε ο φοιτητής στο πρώτο μέρος.

  Στην περίπτωση που επιλέγηκε πρωταρχικός LFSR, η εφαρμογή θα επιλέγει τυχαία κάποιο αλγόριθμο από κατάλογο προκαθορισμένων πρωταρχικών LFSR, οι οποίοι θα πληρούν το κριτήριο της επιλογής του φοιτητή (πχ 10 <= n <= 50).

- Κρυπτανάλυση. Δίνεται το κρυπτογραφημένο μήνυμα και μέρος της αρχής του μηνύματος (γνωστό κείμενο). Η εφαρμογή επιχειρεί επίθεση γνωστού κειμένου (με τη χρήση του αλγόριθμου Berlekamp-Massey) και στην περίπτωση που επιτυγχάνει, δείχνει το αποκρυπτογραφημένο μήνυμα, ενώ στην περίπτωση αποτυχίας ζητά μεγαλύτερο γνωστό κείμενο.



Σχήμα 6.6: Κρυπτογραφία – Επίθεση γνωστού κειμένου

Όλοι οι υπολογισμοί, τα βήματα και η χρήση των αλγορίθμων τα οποία θα εκτελεί η εφαρμογή σε όλα τα βήματα (κρυπτογράφηση, αποκρυπτογράφηση, κρυπτανάλυση, κλπ), θα πρέπει να φαίνονται στη σελίδα. Θα πρέπει ο φοιτητής να μπορεί να ανατρέξει και να βρει τις πράξεις και τα βήματα που χρησιμοποιήθηκαν.

- ***Προστασία***. Τα μέτρα προστασίας και οι καλές πρακτικές που πρέπει να ακολουθούμε σε σχέση με τη χρήση αλγορίθμων ροής.

- ***Βάση Δεδομένων***. Για αυτό το κομμάτι της εφαρμογής θα χρειαστούμε να υλοποιήσουμε στη βάση δεδομένων ένα πίνακα ο οποίος θα περιέχει τη λίστα με τους προκαθορισμένους πρωταρχικούς LFSR. Ο πίνακας θα έχει για κάθε αποθηκευμένο LFSR, την τιμή του n, την περιγραφή του LFSR και την αρχική του κατάσταση.

Πίνακας:	**LFSRsList**
Πεδία:	lfsrID (integer) n (integer) tapsList (string)

Σχήμα 6.7: Κρυπτογραφία – Πίνακας πρωταρχικών LFSR

6.5.4 Webgoat

Το Webgoat είναι μια άλλη εφαρμογή η οποία προσφέρει μαθήματα ασφάλειας, σε προχωρημένο επίπεδο. Αναμένεται από την εφαρμογή μας να περιλαμβάνει ένα θέμα αφιερωμένο στο Webgoat. Το θέμα αυτό θα έχει σκοπό να δώσει το επόμενο βήμα στο φοιτητή που ήδη γνωρίζει τις βασικές αρχές ασφάλειας καθώς και όλες τις κύριες ευπάθειες.

Αυτό το θέμα χρειάζεται να περιλαμβάνει οδηγό εγκατάστασης της εφαρμογής και λεπτομερή οδηγό χρήσης (Walkthrough).

Με αυτό το θέμα, αναμένουμε να δώσουμε λύση στα αρχικά προβλήματα που μπορεί να βρει ο φοιτητής στη χρήση του Webgoat.

6.6 Τεχνολογία

Πιο κάτω παρουσιάζεται η τεχνολογία που χρησιμοποιήθηκε κατά την δημιουργία της εφαρμογής. Παρουσιάζονται τα εργαλεία υλικού και λογισμικού και αναλύονται οι λόγοι επιλογής τους.

6.6.1 Υλικό

Ενώ η ανάπτυξη κώδικα είναι πιο εύκολη και γρήγορη με τα εργαλεία στο περιβάλλον Windows, η εφαρμογή είναι προτιμότερο να εγκατασταθεί και να λειτουργεί σε περιβάλλον Linux. Για αυτό το λόγο χρησιμοποιήθηκαν δύο υπολογιστές κατά τη διάρκεια της υλοποίησης. Ένα υπολογιστή Windows για σκοπούς ανάπτυξης κώδικα και ένα υπολογιστή Linux για σκοπούς φιλοξενίας και λειτουργίας της εφαρμογής (Πίνακας 6.1). Αξιοποιήθηκε επίσης ο εξυπηρετητής του Ανοικτού Πανεπιστημίου Κύπρου (ΑΠΚΥ), από όπου δημοσιεύεται η σελίδα στο διαδίκτυο.

Υλικό (Hardware)	Λειτουργικό σύστημα	Ρόλος στο έργο
Ηλεκτρονικός Υπολογιστής	Windows 7	Development: Συγγραφή και ανάπτυξη κώδικα.
Raspberry Pi Model B	Raspbian 3.18 (Debian)	Εξυπηρετητής: Φιλοξενία και λειτουργία της σελίδας στο τοπικό δίκτυο, για σκοπούς υλοποίησης και ελέγχου.
Εξυπηρετητής ΑΠΚΥ	Linux	Εξυπηρετητής: Δημοσίευση της σελίδας στο διαδίκτυο.

Πίνακας 6.1: Υλικό που χρησιμοποιήθηκε

6.6.2 LAMP

Για την ανάπτυξη της εφαρμογής επιλέγηκε η χρήση της πλατφόρμας LAMP. Η πλατφόρμα LAMP αφορά την ανάπτυξη ιστοσελίδων και αποτελείται από μια ομάδα λογισμικού, το **L**inux (λειτουργικό σύστημα), **A**pache (webserver), **M**ySQL (βάση δεδομένων) και **P**HP (γλώσσα προγραμματισμού). Τα εργαλεία αυτά είναι όλα λογισμικά ανοικτού και ελεύθερου κώδικα, είναι δωρεάν, και λειτουργούν πολύ καλά μεταξύ τους. Η πλατφόρμα LAMP αποτελεί σήμερα την πιο διαδεδομένη πλατφόρμα εφαρμογής και ανάπτυξης ιστοσελίδων, με περισσότερο από τα δύο τρίτα του διαδικτύου να αποτελείται από σελίδες ανεπτυγμένες σε αυτήν [32]. Επιπρόσθετα, τα εργαλεία LAMP είναι εύκολα στη χρήση, με απλή και κατανοητή εγκατάσταση και παραμετροποίηση.

6.6.3 Apache

Ο Apache αποτελεί τον webserver της πλατφόρμας LAMP, συμβατός με το πρωτόκολλο HTTP/1.1. Είναι ευέλικτος, επεκτάσιμος (με τη χρήση πρόσθετων modules), εύκολος στην εγκατάσταση και έχει κατανοητό και ευέλικτο σύστημα παραμετροποίησης. Μπορεί να λειτουργήσει σχεδόν σε όλα τα λειτουργικά συστήματα και σίγουρα σε περιβάλλον Windows ή Linux [02].

Κατά την δημιουργία της εφαρμογής αυτής, χρησιμοποιήθηκε ο Apache Webserver 2.2.22 (Debian).

6.6.4 PHP

Για την υλοποίηση της εφαρμογής επιλέγηκε η γλώσσα προγραμματισμού PHP. Επιπρόσθετα από το ότι η PHP αποτελεί μέρος της πλατφόρμας LAMP, είναι πολύ εύκολη γλώσσα να τη μάθει κάποιος, δίνει μεγάλες επιδόσεις σε χρόνους εκτέλεσης αλγορίθμων, έχει πολλές βιβλιοθήκες διαθέσιμες, έχει μεγάλη κοινότητα χρηστών και αποτελεί λογισμικό ελεύθερο και δωρεάν [36].

Η PHP είναι γλώσσα η οποία εκτελείται στον εξυπηρετητή όπου και μεταφράζεται σε κείμενο HTML πριν αποσταλεί στο φυλλομετρητή του χρήστη.

Μερικές από τις **πιο σημαντικές εντολές και μέθοδοι PHP** οι οποίες χρησιμοποιήθηκαν κατά την υλοποίηση της εφαρμογής αυτής είναι [22], [35], [36]:

include()

Με την εντολή incluce μπορούμε να συμπεριλάβουμε στον κώδικα μας κάποιο άλλο αρχείο php. Έτσι μπορούμε να επαναχρησιμοποιήσουμε τον κώδικα μας. Μία συχνή χρήση της εντολής αυτής στην ανάπτυξη ιστοσελίδων έχει να κάνει με τη δημιουργία ενός αρχείου php για την επικεφαλίδα (header) της ιστοσελίδας μας, το οποίο να περιλαμβάνουμε (inlcude) σε όλες τις σελίδες της εφαρμογής μας, χωρίς να χρειάζεται να το ξαναγράψουμε.

```
<?php
  include('header.php');
?>
```

Χρήση με βάσεις δεδομένων

Η PHP παρέχει εντολές και μεθόδους για να μπορέσει κάποιος να συνδεθεί με κάποια βάση δεδομένων, στην οποία να εκτελεί εντολές SQL (δημιουργία και διαγραφή πίνακα, update, insert, delete, select).

Για χρήση με βάσεις δεδομένων MySQL, χρησιμοποιούμε τις εντολές **mysqli_connect** και **mysqli_query**:

```
<?php
  $myDB = mysqli_connect(host,username,password,db_name);
  if ($myDB)
  {
    $query  = "select * from some_table";
    $result = mysqli_query($myDB, $query);
    if (mysqli_num_rows($result) > 0)
    {
      while($row = $result->fetch_assoc())
      {
        echo $row['table field name'];
      }
    }
  }
?>
```

```
```

hash()

Με την εντολή αυτή, η PHP δημιουργεί το αποτύπωμα ή κατακερματισμένη τιμή (message digest) κάποιου κειμένου, με τη χρήση αλγορίθμου κατακερματισμού hash, οποίος ορίζεται μέσα στην εντολή (πχ sha256):

```
<?php
  $cyphertext = hash("sha256",$cleartext,false);
?>
```

6.6.5 MySQL

Το σύστημα διαχείρισης βάσεων δεδομένων MySQL, είναι το πιο διαδεδομένο μεταξύ των εργαλείων βάσης δεδομένων ανοικτού κώδικα στο διαδίκτυο και χρησιμοποιείται από σελίδες όπως οι Facebook, Google, Adobe [15]. Χρησιμοποιείται πολύ συχνά με την PHP και αποτελεί αναπόσπαστο κομμάτι της πλατφόρμας LAMP. Αποτελείται από λογισμικό το οποίο εγκαθιστούμε στον εξυπηρετητή και το οποίο μπορούμε να διαχειριστούμε με τη χρήση εργαλείων client σε τοπικό επίπεδο.

Για τη διαχείριση της MySQL στο περιβάλλον της εφαρμογής αυτής χρησιμοποιήθηκαν δύο εργαλεία client σε τοπικό επίπεδο:

- phpMyAdmin [23]. Αποτελεί web-based κονσόλα με την οποία μπορεί κάποιος να διαχειριστεί τη βάση δεδομένων του μέσω του φυλλομετρητή. Χρησιμοποιήθηκε αυτή η μέθοδος για πρόσβαση και διαχείριση της βάσης δεδομένων στον εξυπηρετητή του ΑΠΚΥ.

- MySQL Workbench [14]. Είναι η επίσημη εφαρμογή client που συστήνεται από τη MySQL. Χρησιμοποιήθηκε αυτό το εργαλείο κυρίως για διαχείριση της βάσης δεδομένων του εξυπηρετητή στο τοπικό δίκτυο του γράφοντος.

6.7 Υλοποίηση

6.7.1 Εγκατάσταση και παραμετροποίηση του Λειτουργικού Συστήματος

Όπως αναφέρθηκε πιο πάνω, για τον εξυπηρετητή της ιστοσελίδας στο τοπικό δίκτυο χρησιμοποιήθηκε υπολογιστής με λειτουργικό σύστημα Linux. Για την εγκατάσταση και παραμετροποίηση του λειτουργικού του συστήματος ακολουθήσαμε τα πιο κάτω βήματα:

Εγκατάσταση λειτουργικού συστήματος:

Κατεβάζουμε το "αρχείο εικόνας" (image) του λειτουργικού συστήματος Raspbian από την επίσημη σελίδα του raspberry pi [25] και την εγκαθιστούμε σε μια μνήμη-κάρτα τύπου SD.

Τοποθετούμε την κάρτα SD στο raspberry pi και το ενεργοποιούμε.

Παραμετροποίηση λειτουργικού συστήματος:

α) Αλλαγή αρχικού συνθηματικού:

```
sudo passwd pi
reboot
```

β) Ορισμός σταθερής διεύθυνσης IP:

Πρώτα κάνουμε αντίγραφο ασφαλείας του αρχείου `/etc/network/interfaces` και στη συνέχεια το "ανοίγουμε":

```
cp /etc/network/interfaces /etc/network/interfaces.bak
nano /etc/network/interfaces
```

και κάνουμε τις ακόλουθες αλλαγές:

```
auto lo
iface lo inet loopback
auto eth0
iface eth0 inet static
      address 192.168.1.77
      netmask 255.255.255.0
      network 192.168.1.0
    broadcast 192.168.1.255
      gateway 192.168.1.1
```

```

```

Στη συνέχεια επανεκκινούμε την υπηρεσία networking:

```
/etc/init.d/networking restart
```

γ) Κάνουμε όλες τις αναβαθμίσεις στο λειτουργικό σύστημα:

```
apt-get update
apt-get upgrade
```

6.7.2 Εγκατάσταση Apache, PHP και MySQL

Αφού εγκαταστήσουμε και παραμετροποιήσουμε το λειτουργικό μας σύστημα, προχωρούμε στην ολοκλήρωση του περιβάλλοντος LAMP, το οποίο περιγράφηκε πιο πάνω, με την εγκατάσταση των webserver (Apache), scripting preprocessor (PHP) και βάσης δεδομένων (MySQL):

```
apt-get install apache2 php5 libapache2-mod-php5
service apache2 restart
apt-get install mysql-server mysql-client php5-mysql
```

Ελέγχουμε ότι ο εξυπηρετητής μας εγκαταστάθηκε και δουλεύει σωστά ανοίγοντας τον φυλλομετρητή μας στη διεύθυνση ip του apache server. Αναμένουμε να ανοίξει σελίδα με το κείμενο "**It works!**".

6.7.3 Υλοποίηση της Βάσης Δεδομένων

Καθώς η ιστοσελίδα αυτή είναι περισσότερο εφαρμογή προσομοίωσης και επίδειξης παρά εφαρμογή συλλογής και επεξεργασίας δεδομένων, η βάση δεδομένων μας αποτελείται από τρεις μόνο πίνακες, ξεχωριστούς και μη συσχετισμένους μεταξύ τους. Ο κάθε πίνακας εξυπηρετεί ξεχωριστό θέμα στην εφαρμογή.

Πίνακας Users

Ο πίνακας Users χρησιμοποιείται από την εφαρμογή επίδειξης επίθεσης SQL injection. Περιέχει στοιχεία χρηστών τα οποία ο χρήστης της εφαρμογής θα προσπαθήσει να αποσπάσει με επίθεση SQL injection.

Ο κώδικας SQL για τη δημιουργία του πίνακα :

```
create table `Users` (`userID` INT NOT NULL auto_increment,
                      `email` varchar(100) NOT NULL,
                      `username` varchar(100) NOT NULL,
                      `password` VARCHAR(100) NOT NULL,
                       `role` VARCHAR(50) NOT NULL,
                       PRIMARY KEY (`userID`)
                      );
```

Το πεδίο userID ορίζεται πρωτεύον κλειδί με ιδιότητα να λαμβάνει αυτόματα αριθμητικές τιμές με αύξουσα σειρά (auto_increment).

Ορίζουμε όλα τα πεδία NOT NULL έτσι ώστε να μην επιδέχεται ο πίνακας κενά στοιχεία σε κανένα πεδίο.

Επιπρόσθετα από τη δημιουργία του πίνακα, είναι αναγκαίο να εισαγάγουμε μερικά στοιχεία, τα οποία θα χρειαστεί χρησιμοποιήσει η εφαρμογή μας. Για την εισαγωγή μιας εγγραφής στον πίνακα χρησιμοποιούμε την εντολή SQL:

```
insert into `Users` (`email`, `username`, `password`, `role`)
 values ('BenjaminRahman@dayrep.com', 'ben', 'Lara94', 'Normal User');
```

Δεν ορίζουμε τιμή για το πεδίο UserID αφού είναι ήδη ρυθμισμένο να λαμβάνει τιμές αυτόματα.

Ο πλήρης κώδικας SQL που χρησιμοποιήθηκε για την εισαγωγή εγγραφών στον πίνακα Users βρίσκεται στο παράρτημα.

Πίνακας UsersSha

Αυτός ο πίνακας χρησιμοποιείται από την εφαρμογή επίθεσης λεξικού. Περιέχει στοιχεία χρηστών τα οποία ο χρήστης της εφαρμογής θα προσπαθήσει να αποσπάσει με την επίθεση αυτή. Είναι παρόμοιος πίνακας με τον Users, με μόνη διαφορά ότι τα συνθηματικά των χρηστών

σε αυτό τον πίνακα αποθηκεύονται σε μορφή αποτυπώματος (κατακερματισμένα) με τον αλγόριθμο SHA256.

Ο κώδικας SQL για τη δημιουργία του πίνακα :

```
create table `UsersSha` (`userID` INT NOT NULL auto_increment,
                         `email` varchar(100) NOT NULL,
                         `username` varchar(100) NOT NULL,
                         `passwordSha` VARCHAR(64) NOT NULL,
                         `role` VARCHAR(50) NOT NULL,
                         PRIMARY KEY (`userID`)
                        );
```

Το πεδίο userID ορίζεται πρωτεύον κλειδί με ιδιότητα να λαμβάνει αυτόματα αριθμητικές τιμές με αύξουσα σειρά (auto_increment).

Ορίζουμε όλα τα πεδία NOT NULL έτσι ώστε να μην επιδέχεται ο πίνακας κενά στοιχεία σε κανένα πεδίο.

Επιπρόσθετα από τη δημιουργία του πίνακα, είναι αναγκαίο να εισαγάγουμε μερικά στοιχεία, τα οποία θα χρειαστεί χρησιμοποιήσει η εφαρμογή μας. Για την εισαγωγή μιας εγγραφής στον πίνακα χρησιμοποιούμε την εντολή SQL:

```
insert into `UsersSha`
       (`email`, `username`, `passwordSha`, `role`)
values ('AliceGreen@teleworm.us'
      , 'alice'
      ,'e439adedd5a0fc9bdfc486833803a27fac2b7f845cfcb4dd6f0f0efe19ae7f6e'
      , 'Administrator');
```

Δεν ορίζουμε τιμή για το πεδίο UserID αφού είναι ήδη ρυθμισμένο να λαμβάνει τιμές αυτόματα.

Ο πλήρης κώδικας SQL που χρησιμοποιήθηκε για την εισαγωγή εγγραφών στον πίνακα Users βρίσκεται στο παράρτημα.

Πίνακας LFSRsList

Ο πίνακας αυτός περιέχει λίστα με προκαθορισμένους πρωταρχικούς LFSR. Η εφαρμογή "Κρυπτογραφία – Επίθεση γνωστού μηνύματος" χρησιμοποιεί τον πίνακα για να επιλέξει τυχαία

κάποιο LFSR, και να τον χρησιμοποιήσει στην παραγωγή κλειδοροής κατά τη φάση της κρυπτογράφησης μηνύματος.

Ο κώδικας SQL για τη δημιουργία του πίνακα :

```
create table `LFSRsList` (`lfsrID` INT NOT NULL auto_increment,
                          `n` int NOT NULL,
                          `tapsList` VARCHAR(250) NOT NULL,
                            PRIMARY KEY (`lfsrID`)
                          );
```

Το πεδίο lfsrID ορίζεται πρωτεύον κλειδί με ιδιότητα να λαμβάνει αυτόματα αριθμητικές τιμές με αύξουσα σειρά (auto_increment).

Ορίζουμε όλα τα πεδία NOT NULL έτσι ώστε να μην επιδέχεται ο πίνακας κενά στοιχεία σε κανένα πεδίο.

Για τους πρωταρχικούς LFSR που εισάγαμε στον πίνακα, χρησιμοποιήσαμε τον πίνακα πρωταρχικών πολυωνύμων του Zivkovic [37].

Για να αναπαραστήσουμε στον πίνακα μας ένα πρωταρχικό LFSR, χρησιμοποιούμε το πεδίο tasplist. Το πεδίο αυτό περιέχει τις βαθμίδες ανάδρασης του LFSR, από δεξιά προς αριστερά. Δεν ορίζουμε την αρχική κατάσταση η οποία αφήνεται να οριστεί από την υλοποίηση του αλγόριθμου.

Έτσι, για τον LFSR με ανάδραση στις βαθμίδες 1, 4, 10 και 11 (από δεξιά προς αριστερά), θα βάζαμε στο πεδίο tapsList το κείμενο "1,4,10,11". Ο λόγος που επιλέξαμε να αποθηκεύσουμε απευθείας τις βαθμίδες ανάδρασης του LFSR, και όχι το πολυώνυμο από το οποίο προκύπτουν, είναι για να διευκολύνουμε τον κώδικα που θα χρησιμοποιήσει το πολυώνυμο αυτό κατά τη διαδικασία της κρυπτογράφησης. Με αυτό τον τρόπο ο αλγόριθμος είναι πολύ πιο γρήγορος αφού δεν χρειάζεται να μετασχηματίσει το πολυώνυμο σε βαθμίδες ανάδρασης κάθε φορά που θα το χρησιμοποιεί. Εξάλλου, οι βαθμίδες ανάδρασης ορίζουν μονοσήμαντα το αντίστοιχο πολυώνυμο και αντίστροφα. Με αυτόν τον τρόπο, έχουμε τις βαθμίδες έτοιμες. Δεν μας απασχολεί σε αυτό το σημείο η χρήση μιας κωδικοποίησης πιο κατανοητής από την επιστημονική κοινότητα επειδή ο κατάλογος αυτός αποτελεί εσωτερικό κομμάτι της εφαρμογής και δεν θα είναι ορατός από τρίτους.

Για τη μετατροπή των πολυωνύμων σε βαθμίδες ανάδρασης χρησιμοποιήθηκε το ο πιο κάτω κώδικας στη λίστα των πολυωνύμων:

```
tapsList = "1" + ","
         + (F1+1) + ","
         + (F2+1) + ","
         + (F3+1) + ","
         + (F4+1) + ","
         + (F5+1)
```

όπου F1..F5, οι συντελεστές του πολυώνυμου.

Για την εισαγωγή των προκαθορισμένων πρωταρχικών γεννητριών LFSR, δημιουργήσαμε ένα αρχείο δεδομένων κειμένου (tab delimited). Το αρχείο περιέχει τα στοιχεία n και tapsList για κάθε πρωταρχικό LFSR που θα μπει στον πίνακα. Στη συνέχεια φορτώσαμε το αρχείο στον πίνακα με την εντολή SQL:

```
truncate table LFSRsList;

  load data infile 'LFSRList.txt'
  into table LFSRsList
fields terminated by '\t'
 lines terminated by '\n'
ignore 1 rows (n, tapsList, @dummy);
```

6.7.4 Υλοποίηση της κυρίως ιστοσελίδας

Το σχέδιο της σελίδας μας και η γενική της διάταξη καθορίστηκε με γνώμονα τις πιο κάτω απαιτήσεις:

- **Επεκτάσιμο**. Να μπορούμε εύκολα να προσθέσουμε όσα θέματα θέλουμε.

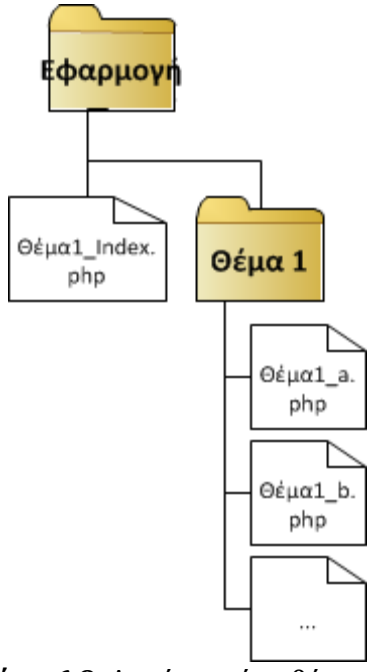
- **Προγραμματιστικά διαχειρίσιμο**. Να μην έχουμε επαναλαμβανόμενο κώδικα στις σελίδες. Κοινά θέματα σε όλες τις σελίδες να είναι μόνο μία φορά υλοποιημένα.

- **Απλό**. Ο χρήστης να μπορεί εύκολα να πλοηγηθεί μεταξύ των θεμάτων χωρίς να δημιουργείται σύγχυση.

Καταλήξαμε λοιπόν στη διάταξη του σχήματος 6.8.

Σχήμα 6.8: Διάταξη ιστοσελίδας

Για το κάθε θέμα αναπτύχθηκε ξεχωριστή/αυτόνομη ιστοσελίδα, η οποία καλείται από τη λίστα θεμάτων στην επικεφαλίδα και ενσωματώνεται στο κυρίως μέρος της εφαρμογής. Επιπρόσθετα αναπτύχθηκαν σε ξεχωριστές σελίδες η επικεφαλίδα και ο υπότιτλος. Συνεπώς, για κάθε θέμα της εφαρμογής μας δημιουργήσαμε τη δομή αρχείων του σχήματος 6.9.



Σχήμα 6.9: Δομή αρχείων θέματος

Ο κώδικας php του αρχείου ***Θέμα1_index.php*** αξιοποιεί τη συνάρτηση ***include()*** για να ενσωματώσει το κυρίως περιεχόμενο του θέματος μαζί με την επικεφαλίδα και τον υπότιτλο:

```
<body>
<table width=100% border="0" cellpadding="0" cellspacing="0">
   <tr>
    <td>
    <?php include('header.php'); ?>
    </td>
  </tr>
  <tr>
    <table width=100% border="0" cellpadding="0" cellspacing="0">
    <tr>
    <td width=5%><td>
    <td width=90%><?php include('θέμα1/θέμα1_content.php'); ?></td>
    <td width=5%><td>
    </tr>
    </table>
  </tr>
  <tr>
    <td>
    <hr color="#CCCCCC">
    <?php include('footer.php'); ?>
    </td>
  </tr>
</table>
</body>
```

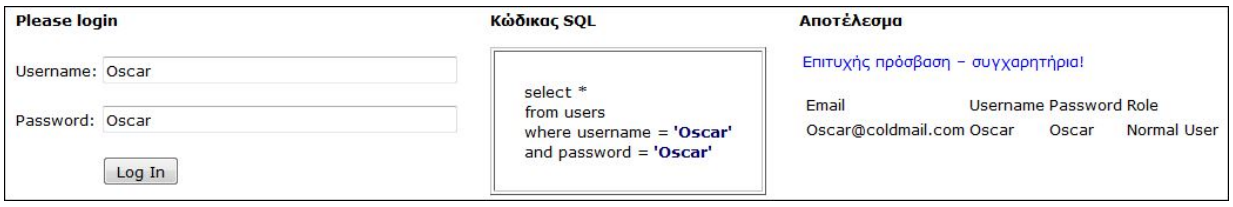
6.7.5 Υλοποίηση απλής εφαρμογής Login για SQL Injection

Στην ενότητα "Επίδειξη Επίθεσης" του θέματος ***SQL Injection***, αναπτύχθηκε μια απλή εφαρμογή Login η οποία είναι εσκεμμένα ευάλωτη σε επίθεση SQL Injection. Η εφαρμογή αποτελείται από τρία μέρη:

- **Login**. Περιέχει δύο κουτιά κειμένου. Ένα για όνομα χρήστη και ένα για το συνθηματικό. Υπάρχει επίσης ένα κουμπί εντολής το οποίο εκτελεί την εντολή SQL στη βάση δεδομένων.

- **Κώδικας SQL**. Δείχνει σε μορφή κειμένου, τον κώδικα SQL που εκτελείται στη βάση δεδομένων.

- **Αποτέλεσμα**. Παρουσιάζει το αποτέλεσμα της πράξης **Login**.



Σχήμα 6.10: Η εφαρμογή Login για SQL injection

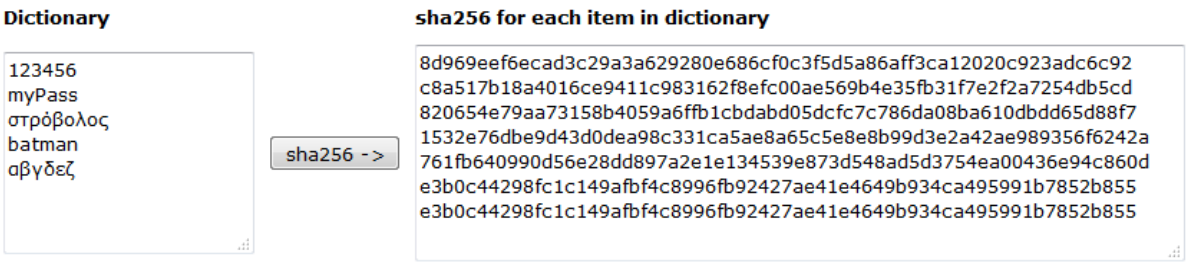
Η εφαρμογή είναι ευάλωτη σε επίθεση SQL injection, επειδή το κουμπί εντολής Login, χρησιμοποιεί τα δεδομένα που θέτει ο χρήστης (Username και Password) χωρίς να επικυρωθούν. Κτίζεται η εντολή SQL με απλή σύνθεση κομματιών κειμένου:

```
$query = "select * from UsersTemp where username = '"
          .$username
          ."' and password = '"
          .$password
          ."'";
```

Χρειάζεται να είμαστε πολύ προσεκτικοί όταν αφήνουμε την εφαρμογή μας ανοικτή σε επίθεση SQL injection. Ο χρήστης της εφαρμογής μας είναι σε θέση να πειραματιστεί με όλες τις εντολές SQL, οι οποίες μπορεί να δημιουργήσουν μόνιμο πρόβλημα στη βάση δεδομένων μας και συνεπώς να μην μπορεί να λειτουργήσει η εφαρμογή. Μια τέτοια ακραία περίπτωση είναι να χρησιμοποιήσει κάποιος την εντολή **drop** όπου και θα ***διαγραφεί μόνιμα*** ο πίνακας μας από τη βάση δεδομένων. Για να προστατεύσουμε την εφαρμογή μας από αυτό τον κίνδυνο, δημιουργούμε ***προσωρινό*** πίνακα **users** κάθε φορά που πατά ο χρήστης το κουμπί Login, και εκτελούμε την εντολή SQL που προκύπτει στον προσωρινό αυτό πίνακα. Εξασφαλίζουμε έτσι ότι ο πίνακας επί του οποίου θα εκτελεστεί η εντολή SQL υπάρχει πάντα και στη δομή που τον θέλουμε. Στο όνομα του προσωρινού πίνακα χρησιμοποιούμε ένα session id, έτσι ώστε να μπορεί να χρησιμοποιηθεί μόνο από το συγκεκριμένο χρήστη.

6.7.6 Υλοποίηση της συνάρτησης κατακερματισμού SHA256

Κατά τη διάρκεια εκτέλεσης του σεναρίου επίθεσης λεξικού, ο χρήστης καλείται να δημιουργήσει το αποτύπωμα των πιθανών συνθηματικών που συνέταξε (λεξικό) με τη χρήση της συνάρτησης κατακερματισμού sha256.



Σχήμα 6.11: Χρήση του αλγόριθμου κατακερματισμού SHA256

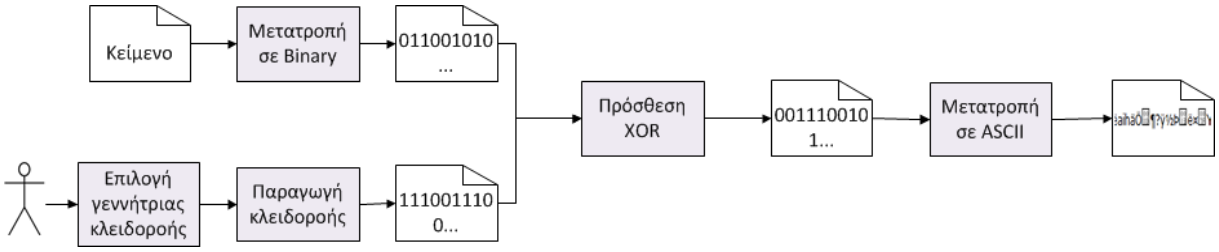
Ο χρήστης τοποθετεί τις λέξεις του λεξικού στο κουτί κειμένου "Dictionary" και πατά το κουμπί "sha256 ->". Η εφαρμογή δημιουργεί το αποτύπωμα όλων των λέξεων και το τοποθετεί στο κουτί "sha256".

Για την κρυπτογράφηση χρησιμοποιήθηκε η αντίστοιχη μέθοδος από τις βιβλιοθήκες της PHP, hash:

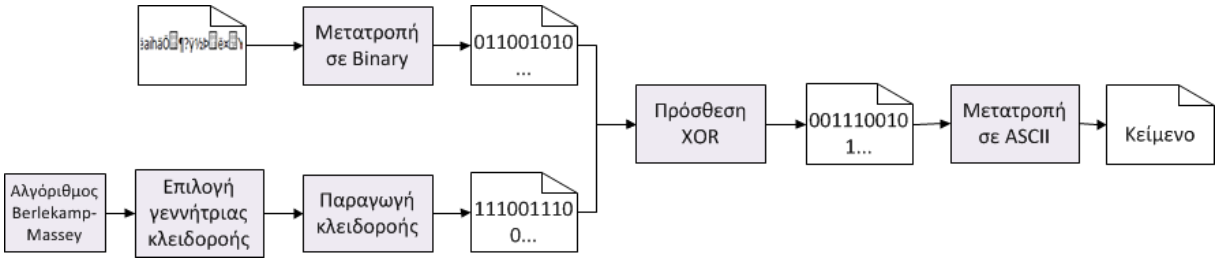
```
$cyphertext = $cyphertext.hash("sha256",$cleartext,false);
```

6.7.7 Παραγωγή κλειδοροής από γεννήτρια LFSR και υλοποίηση κρυπτογράφησης / αποκρυπτογράφησης με αλγόριθμο ροής

Για τις ανάγκες της επίδειξης επίθεσης γνωστού μηνύματος, η εφαρμογή καλείται να κρυπτογραφήσει και αποκρυπτογραφήσει, χρησιμοποιώντας αλγόριθμο ροής.



Σχήμα 6.12: Κρυπτογράφηση με LFSR (αλγόριθμος ροής)



Σχήμα 6.13: Αποκρυπτογράφηση με LFSR (αλγόριθμος ροής)

Η κρυπτογράφηση και αποκρυπτογράφηση με αλγόριθμους ροής δεν διαφέρουν μεταξύ τους προγραμματιστικά (όπως φαίνεται και στα σχήματα). Οι μόνες διαφορές είναι:

- Στην κρυπτογράφηση αρχίζουμε με το κείμενο και καταλήγουμε στο κρυπτοκείμενο ενώ στην αποκρυπτογράφηση αρχίζουμε με το κρυπτοκείμενο και καταλήγουμε στο ανοικτό κείμενο.

- Στην κρυπτογράφηση χρησιμοποιείται η γεννήτρια κλειδοροής που επιλέγει ο χρήστης, ενώ στην αποκρυπτογράφηση η γεννήτρια κλειδοροής δεν είναι γνωστή στην εφαρμογή αλλά την "μαντεύει" ο αλγόριθμος Berlekamp – Massey με επίθεση γνωστού κειμένου.

Υλοποιήθηκαν λοιπόν οι μέθοδοι που χρειάζονται για την κρυπτογράφηση, οι οποίες χρησιμοποιήθηκαν και κατά την αποκρυπτογράφηση:

Μετατροπή σε Binary:

Για τη μετατροπή κειμένου ASCII σε δυαδικό κείμενο (0,1), υλοποιήθηκε η μέθοδος **characterToBinary($m)**. Η μέθοδος αυτή λαμβάνει ως παράμετρο το κείμενο σε μορφή ASCII και επιστρέφει το αντίστοιχο κείμενο στο δυαδικό σύστημα.

Η διαδικασία η οποία ακολουθείται είναι:

1. Μετατροπή κειμένου ASCII σε πίνακα χαρακτήρων (array). Χρήση της μεθόδου PHP str_split().

2. Μετατρέπουμε κάθε χαρακτήρα του πίνακα στον αντίστοιχο δεκαδικό αριθμό του από τον πίνακα ASCII (εντολή PHP ord()) και στη συνέχεια από τον δεκαδικό αριθμό μετατρέπουμε στον αντίστοιχο δυαδικό (εντολή PHP decbin()). Χρησιμοποιούμε δυαδικούς αριθμούς οκτώ χαρακτήρων (προσθέτουμε μηδενικά στα αριστερά σε περίπτωση που έχουμε μικρούς αριθμούς – εντολή PHP str_pad())

3. Επιστρέφουμε σε μορφή ενιαίου κειμένου τις δυαδικές αναπαραστάσεις των χαρακτήρων του αρχικού μηνύματος

Για σκοπούς απλοποίησης της υλοποίησης, υιοθετήθηκε η σύμβαση ότι κρυπτογραφούμε κείμενο το οποίο αποτελείται μόνο από χαρακτήρες του αγγλικού αλφαβήτου και αριθμούς.

Μετατροπή σε ASCII

Για τη μετατροπή κάποιας δυαδικής αλληλουχίας σε χαρακτήρες ASCII, υλοποιήθηκε η μέθοδος **binaryToCharacter($b)**. Η μέθοδος αυτή λαμβάνει ως παράμετρο το κείμενο σε δυαδική μορφή και επιστρέφει το αντίστοιχο κείμενο ASCII.

Η διαδικασία η οποία ακολουθείται είναι ακριβώς αντίστροφη με τη διαδικασία που ακολουθήσαμε κατά τη μετατροπή σε Binary (μέθοδος characterToBinary()).

Παραγωγή κλειδοροής

Για την παραγωγή της κλειδοροής, υλοποιήθηκε η μέθοδος **getNextState()**. Λαμβάνει ως παραμέτρους:

- Την κατάσταση της γεννήτριας LFSR, σε μορφή πίνακα (array) μηδενικών και άσσων (0,1). Για παράδειγμα, LFSR μήκους 5 και αρχική κατάσταση 00101, θα δοθεί ο πίνακας 0,0,1,0,1.

  Επειδή οι LFSR που χρησιμοποιούμε είναι πρωταρχικοί, δεν υπάρχει οποιαδήποτε απαίτηση ή περιορισμός για την αρχική κατάσταση. Έτσι υιοθετήθηκε στην εφαρμογή αυτή να δίνεται αρχική κατάσταση η διάταξη των βαθμίδων ανάδρασης.

- Τις βαθμίδες ανάδρασης του LFSR, σε μορφή κειμένου, όπως είναι αποθηκευμένες στη βάση δεδομένων.

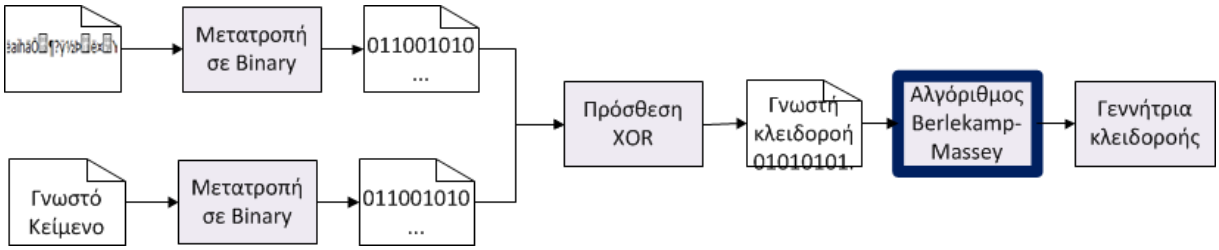
Η μέθοδος επιστρέφει την κατάσταση του LFSR μετά από μία ανάδραση. Μπορούμε έτσι να πάρουμε το δεξιότερο στοιχείο της κατάστασης LFSR που επιστρέφεται και να το χρησιμοποιήσουμε στην κρυπτογράφηση/αποκρυπτογράφηση.

Πρόσθεση XOR

Υλοποιήθηκε απλή μέθοδος η οποία λαμβάνει δύο δυαδικά ψηφία, τα προσθέτει με XOR και επιστρέφει το αποτέλεσμα σε μορφή επίσης δυαδικού ψηφίου.

6.7.8 Υλοποίηση αλγόριθμου Berlekamp – Massey για την εφαρμογή "Κρυπτογραφία – Επίθεση γνωστού μηνύματος"

Ο αλγόριθμος Berlekamp – Massey υλοποιήθηκε σε PHP (από τη αρχή) για τους σκοπούς αυτής της εφαρμογής. Για την υλοποίηση του βασιστήκαμε στον ψευδοκώδικα όπως παρουσιάζεται από τους Menezes et al ([12]) αλλά και σε υλοποιήσεις Java και C ([31]).

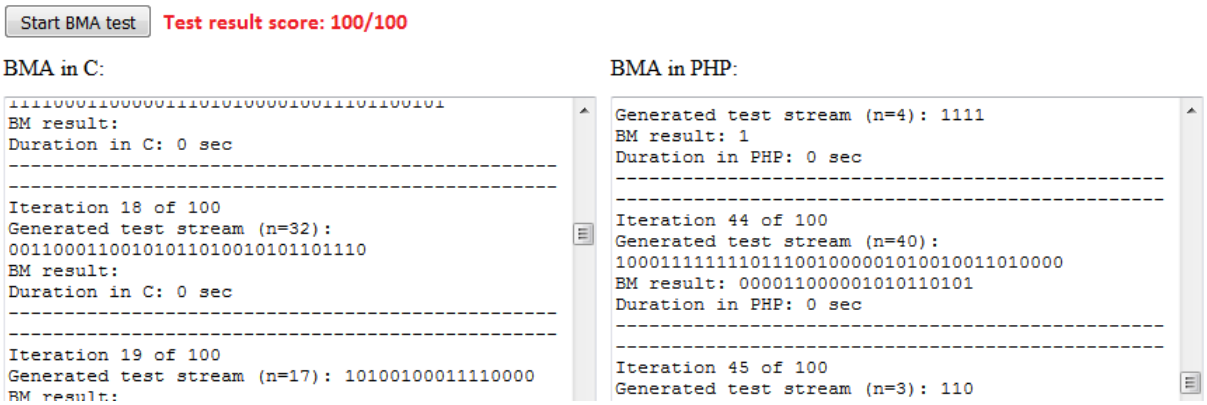


Σχήμα 6.14: Χρήση του αλγόριθμου Berlekamp – Massey

Η μέθοδος που υλοποιήθηκε (**BMA($stream)**) λαμβάνει ως παράμετρο τη γνωστή κλειδοροή και επιστρέφει τον μικρότερο πρωταρχικό LFSR που μπορεί να την παράγει. Η εφαρμογή λαμβάνει από το χρήστη το κρυπτοκείμενο και μέρος του αρχικού μηνύματος (γνωστό κείμενο). Με αυτά τα δεδομένα υπολογίζει το κομμάτι της κλειδοροής που αντιστοιχεί στην κρυπτογράφηση του γνωστού κειμένου (γνωστή κλειδοροή) και μετά καλεί τη μέθοδο BMA() με παράμετρο τη γνωστή κλειδοροή. Το αποτέλεσμα του αλγόριθμου Berlekamp – Massey επικυρώνεται και εάν είναι ικανοποιητικό, η εφαρμογή προχωρά στην αποκρυπτογράφηση, αλλιώς ζητά από το χρήστη να ξαναδοκιμάσει με πιο μεγάλο γνωστό κείμενο.

Η υλοποίηση του αλγορίθμου αυτού είναι αρκετά περίπλοκη, με αναδρομικές κλείσεις σε υπο-μεθόδους και αρκετές αντιγραφές πινάκων μεταξύ τους. Για το λόγο αυτό ήταν αναγκαίο ο έλεγχος της υλοποίησης αυτής να γίνει πολύ πιο συστηματικός και λεπτομερής από τον έλεγχο που έγινε στις υπόλοιπες υλοποιήσεις.

Ο έλεγχος αυτός περιλάμβανε την υλοποίηση μιας **εφαρμογής ελέγχου** η οποία δημιουργεί τυχαίες σειρές γνωστής κλειδοροής και στη συνέχεια καλεί παράλληλα δύο υλοποιήσεις του αλγορίθμου (την υλοποίηση της εφαρμογής σε PHP – που ελέγχεται – και την υλοποίηση σε C – που θεωρείται σωστή) και συγκρίνει τα αποτελέσματα. Η διαδικασία επαναλαμβάνεται εκατοντάδες φορές.



Σχήμα 6.15: Εφαρμογή ελέγχου υλοποίησης Berlekamp – Massey

Κεφάλαιο 7
Επίλογος

Αντικείμενο της παρούσας διατριβής ήταν η ανάπτυξη ενός εικονικού εργαστηρίου για τη διδασκαλία ακαδημαϊκών μαθημάτων στο χώρο της ασφάλειας ή/και κρυπτογραφίας. Η εφαρμογή που αναπτύχθηκε προς το σκοπό αυτό κάλυψε θέματα που αποτελούν κρίσιμους τομείς για την ασφάλεια πληροφοριών, όπως την κρυπτογραφία (συμμετρική κρυπτογράφηση), την ανίχνευση συνθηματικών και τις επιθέσεις τύπου «έγχυσης» SQL (SQL injection). Τα εν λόγω ζητήματα αναπτύχθηκαν επίσης και ως προς το θεωρητικό τους υπόβαθρο. Απώτερος στόχος είναι η αξιοποίηση της εν λόγω εφαρμογής στη διδασκαλία σχετικών Θεματικών Ενοτήτων στο Ανοικτό Πανεπιστήμιο Κύπρου.

Παρόλο που η εφαρμογή αυτή υλοποιήθηκε επαρκώς σε σχέση με τον αρχικό μας σχεδιασμό, και ικανοποιεί πλήρως αυτά τα οποία αναμέναμε πριν αρχίσουμε τη διατριβή αυτή, δεν κλείνει εδώ ο κύκλος ζωής της και σίγουρα δεν θα "μπει στο ράφι".

Τα σχέδια που υπάρχουν για το επόμενο βήμα είναι να χρησιμοποιηθεί η εφαρμογή στην πράξη κατά την ακαδημαϊκή διαδικασία, δηλαδή να δοθεί η σε φοιτητές που ακολουθούν την κατεύθυνση της ασφάλειας αλλά και στους καθηγητές τους, για να αξιοποιηθεί και, ακολούθως, να αξιολογηθεί. Με βάση την αξιολόγηση αυτή θα ληφθούν αποφάσεις όσον αφορά τον εμπλουτισμό του υλικού και τυχόν άλλες τροποποιήσεις (π.χ. στη διεπαφή της με τους χρήστες).

Αυτός εξάλλου ήταν και ο κυριότερος λόγος ο οποίος η εφαρμογή σχεδιάστηκε και υλοποιήθηκε με τρόπο τέτοιο ώστε να είναι επεκτάσιμη – να είμαστε δηλαδή σε θέση να προσθέτουμε εύκολα και άλλα θέματα τα οποία θα κριθούν ως σημαντικά για τους σκοπούς του διδάσκοντα που θα την αξιοποιήσει ή και θέματα τα οποία οι φοιτητές θα θεωρήσουν ότι χρειάζονται εξασκηθούν πρακτικά επάνω τους.

Θεωρούμε ότι η ευκολία με την οποία μπορεί κάποιος να προσθέτει νέες ενότητες στην εφαρμογή, θα βοηθήσει πολύ στο να εμπλουτιστεί κατά το επόμενο ακαδημαϊκό έτος με ακόμη περισσότερες ενότητες, και να εξελιχθεί από εφαρμογή των τριών ενοτήτων που υλοποιήθηκαν κατά τη διάρκεια μιας διατριβής, σε εφαρμογή πολύ περισσότερων.

Βιβλιογραφία

[01] A. Alvare. "How Crackers Crack Passwords or What Passwords to Avoid". Proceedings, UNIX Security Workshop II. 1990.

[02] ASF. "The Apache Software Foundation". 2015; Available from: http://apache.org/. (accessed 29 Oct 2014).

[03] AWASP. "OWASP WebGoat Project". 2015; Available from: www.owasp.org/index.php/Category:OWASP_WebGoat_Project.

[04] CERT. "Malicious HTML Tags Embedded in Client Web Requests". 2000; Available from: https://www.cert.org/historical/advisories/CA-2000-02.cfm? (accessed 30 Dec 2014).

[05] CWE. "2011 CWE/SANS Top 25 Most Dangerous Software Errors". 2011; Available from: http://cwe.mitre.org/top25/. (accessed 12 Mar 2015).

[06] D. Endler. "The evolution of Cross-Site Scripting Attacks". 2002.

[07] European Commission. "Proposal for the EU Data Protection Regulation". 2012.

[08] European Union. "Directive 95/46/EC on the protection of individuals with regards to the processing of personal data and on the free movement of such data". Official Journal of the European Union, n L 281, 23 November 1995, p. 31-50.

[09] S. Friedl. "SQL Injection Attacks by Example". 2007; Available from: http://www.unixwiz.net/techtips/sql-injection.html. (accessed 9 Apr 2015).

[10] A. Huth, M. Orlando, L. Pesante. "Password Security, Protection, and Management". 2012: US-CERT.

[11] M. McDowell, S. Hernan, J.Rafail. "Choosing and Protecting Passwords". 2009: US-CERT.

[12] A. J. Menezes, P. C. van Oorschot, S. A. Vanstone. "Handbook of applied cryptography". 5th ed. 2001.

[13] mganss. "HtmlSanitizer". 2015; Available from: https://github.com/mganss/HtmlSanitizer.

[14] Oracle. "MySQL Workbench". 2015; Available from: www.mysql.com/products/workbench.

[15] Oracle. "Why MySQL?". 2015; Available from: www.mysql.com/why-mysql. (accessed 7 Feb 2015).

[16] OWASP. "Password length & complexity". 2013; Available from: https://www.owasp.org/index.php/Password_length_%26_complexity. (accessed 18 Aug. 2015).

[17] OWASP. "Top 10 2013". 2014; Available from: https://www.owasp.org/index.php/Top_10_2013-Top_10. (accessed 9 Apr 2015).

[18] OWASP. "XSS (Cross Site Scripting) Prevention Cheat Sheet". 2015; Available from: https://www.owasp.org/index.php/XSS_%28Cross_Site_Scripting%29_Prevention_Cheat_Sheet. (accessed 2 Jan 2015).

[19] R. R. Panko. "Corporate computer and Network Security". Second ed. 2010.

[20] W. Patterson, J. Strickland. "Garbage In / Garbage Out: Evaluating Computer Software". The English Record. 1986. 11-15.

[21] C. P. Pfleeger, S. L. Pfleeger. "Security in computing". 4 ed. 2007.

[22] PHP Group. "PHP Manual". 2015; Available from: https://secure.php.net/manual. (accessed 17 Aug 2015).

[23] phpMyAdmin. "phpMyAdmin". 2015; Available from: www.phpmyadmin.net. (accessed 14 Feb 2015).
[24] Ponemon Institute. "2014: A Year of Mega Breaches". January 2015.
[25] RaspberryPi. "Software for the Raspberry Pi". Available from: https://www.raspberrypi.org/downloads/. (accessed 20 Νοε 2014).
[26] R. L. Rivest. "The RC4 encryption algorithm". March 1992: RSA Data Seurity Inc.
[27] SANS Institute. "Password Protection Policy". 2014.
[28] K. Scarfone, M. Souppaya. "Guide to Enterprise Password Management (Draft) ". 2009.
[29] C. E. Shannon. "Communication theory of secrecy systems". Bell System Technical Journal, 1949. **28**(4): p. 656-715.
[30] W. Stallings. "Cryptography and Network Security: Principles and Practice". 6th ed. 2014.
[31] M. Stampar. "Implementation of Berlekamp-Massey Algorithm". 2005. (accessed 28 Dec 2014).
[32] T. Steidler-Dennison. "LAMP lights the web". Redhat Magazine. 2005.
[33] THN. "The Hacker News". 2015; Available from: http://thehackernews.com. (accessed 23 Jul 2015).
[34] H.C.A. van Tilborg, S. Jajodia. "Encyclopedia of Cryptography and Security". 2011.
[35] W3Schools. "PHP 5 Tutorial". 2015; Available from: http://www.w3schools.com/php. (accessed 17 Aug 2015).
[36] L. Welling and L. Thomson. "PHP and MySQL Web Development". Fourth ed. 2009.
[37] M. Zivkovic. "A Table of Primitive Binary Polynomials". Mathematics of computation. Vol. 62. 1994. 385-386.
[38] Α. Γιαννούλας. "Εκπαιδευτικό λογισμικό. Διδακτική αξιοποίηση στο σύγχρονο ψηφιακό περιβάλλον". 2010, Αθήνα: Καυκάς.
[39] Ελληνική Αρχή Προστασίας Δεδομένων Προσωπικού Χαρακτήρα. "Απόφαση ΑΡ.59/2012". 2012.
[40] Π. Θεοδωρόπουλος. "Είδη, χρήση και αξιολόγιση εκπαιδευτικού λογισμικού". Available from: http://www.p-theodoropoulos.gr/ergasies/didakt-ekplogism.pdf. (accessed 11 Απρ 2015).
[41] ΙΤΥ. "Περί Εκπαιδευτικού Λογισμικού". 2003; Available from: http://ecourse.uoi.gr/pluginfile.php/3897/mod_resource/content/0/Xalkidis/Peri_ekpaideytikou_logismikou.pdf. (accessed 11 Απρ. 2015).
[42] Β. Κάτος, Γ. Στεφανίδης. "Τεχνικές κρυπτογραφίας και κρυπτανάλυσης". 2003.
[43] Κ. Λιμνιώτης. "Τεχνικές Επεξεργασίας Σήματος στην Κρυπτογραφία". 2007.

Παράρτημα Α
Αρχικοποίηση της βάσης δεδομένων

Α.1 Εισαγωγή δεδομένων στον πίνακα Users

```
truncate table Users;

insert into `Users` (`email`, `username`, `password`, `role`)
 values ('AliceGreen@teleworm.us', 'alice', 'Gray150581',
         'Administrator');

insert into `Users` (`email`, `username`, `password`, `role`)
 values ('SeanConnolly@jourrapide.com', 'SeanConolly', 'HJK785',
         'Administrator');

insert into `Users` (`email`, `username`, `password`, `role`)
 values ('BenjaminRahman@dayrep.com', 'ben', 'Lara94', 'Normal User');

insert into `Users` (`email`, `username`, `password`, `role`)
 values ('KianClark@armyspy.com', 'Clark', '95358486', 'Normal User');

insert into `Users` (`email`, `username`, `password`, `role`)
 values ('LucaHart@jourrapide.com', 'Luca', 'Oliver', 'Normal User');

insert into `Users` (`email`, `username`, `password`, `role`)
 values ('Morgan@teleworm.us', 'Moran', '12125', 'Normal User');
```

```
insert into `Users` (`email`, `username`, `password`, `role`)
 values ('Long@teleworm.us', 'Long', '100595', 'Normal User');

insert into `Users` (`email`, `username`, `password`, `role`)
 values ('Danielle.Bird@jourrapide.com', 'Danny', 'Nicholas',
         'Normal User');

insert into `Users` (`email`, `username`, `password`, `role`)
 values ('George@coldmail.com', 'George', 'POE493P', 'Normal User');

insert into `Users` (`email`, `username`, `password`, `role`)
 values ('Oscar@coldmail.com', 'Oscar', 'Oscar', 'Normal User');
```

A.2 Εισαγωγή δεδομένων στον πίνακα UsersSha:

```
truncate table UsersSha;

insert into `UsersSha` (`email`, `username`, `passwordSha`, `role`)
  values('AliceGreen@teleworm.us'
        ,'alice'
        ,'e439adedd5a0fc9bdfc486833803a27fac2b7f845cfcb4dd6f0f0efe19ae7f6e'
        ,'Administrator');

insert into `UsersSha` (`email`, `username`, `passwordSha`, `role`)
  values ('SeanConnolly@jourrapide.com'
       ,'SeanConolly'
       ,'4cef69034508fee5fd99c8ddbae5df7b39cffe39fb9fe7b783a4bdf0a9fe873a'
       ,'Administrator');

insert into `UsersSha` (`email`, `username`, `passwordSha`, `role`)
  values ('BenjaminRahman@dayrep.com'
        , 'ben'
        , 'dcf19e1deab6ce1d40aa4836e5a71b028711195852ad630a1209a0fcc7981984'
        , 'Normal User');

insert into `UsersSha` (`email`, `username`, `passwordSha`, `role`)
  values ('KianClark@armyspy.com'
         ,'Clark'
         ,'4141a32aa262b089a958cddc9139bfb79778201d00978afeffa4d937ba08cced'
         ,'Normal User');

insert into `UsersSha` (`email`, `username`, `passwordSha`, `role`)
  values ('LucaHart@jourrapide.com'
         ,'Luca'
         ,'7efa869d0364eea0cd0106a2ef4d1ae9eaec58fe62928c3f1af8fa8da9204ea0'
         ,'Normal User');

insert into `UsersSha` (`email`, `username`, `passwordSha`, `role`)
  values ('Morgan@teleworm.us'
         ,'Moran'
         ,'6932b332cb6f58b5d8aa5625a753544c59897d7a94ae9d373642eb83e
f015b28'
         ,'Normal User'  );
```

```
insert into `UsersSha` (`email`, `username`, `passwordSha`, `role`)
  values ('Long@teleworm.us'
         ,'Long'
         ,'21b4228782ea3890f51461d1079de332d2ac146a8c98a60bc1f83074288371ab'
         ,'Normal User');

insert into `UsersSha` (`email`, `username`, `passwordSha`, `role`)
  values ('Danielle.Bird@jourrapide.com'
         ,'Danny'
         ,'1360974bb4ede1cdbb00371e03951e3fb0cbccef65eaa03cd1e5a379fd4f40f5'
         ,'Normal User'  );

insert into `UsersSha` (`email`, `username`, `passwordSha`, `role`)
  values ('George@coldmail.com'
         ,'George'
         ,'9d53ed67d01edee6046399e6e852ffa3b702b0df09c1f949fb9468181ba83ad1'
         ,'Normal User'  );

insert into `UsersSha` (`email`, `username`, `passwordSha`, `role`)
  values ('Oscar@coldmail.com'
         ,'Oscar'
         ,'ca66a852a9e96c40f4cce7972d994914909b646b2564e8d25dd4003656b3dd63'
         ,'Normal User'  );
```

Παράρτημα Β
Σύντομη περιγραφή του εργαλείου WebGoat

Το εργαλείο WebGoat είναι μια σελίδα (επίτηδες) ευάλωτη σε γνωστές επιθέσεις ασφαλείας και έχει εκπαιδευτικό χαρακτήρα. Στην κύρια σελίδα, έχουμε μια λίστα από τις ευπάθειες στις οποίες η σελίδα είναι ευάλωτη. Οι ευπάθειες παρουσιάζονται στη σελίδα σε μορφή μαθημάτων (Lessons). Ο χρήστης επιλέγει την ευπάθεια την οποία επιθυμεί να δοκιμάσει και να μάθει και η σελίδα ανοίγει το αντίστοιχο Lesson.

Σε κάθε μάθημα, ο χρήστης βλέπει μερικά λόγια για την ευπάθεια και πώς λειτουργεί η ευπάθεια (Lesson plan). Επίσης έχει τη δυνατότητα να δοκιμάσει επίθεση στη σελίδα της ευπάθειας, να δει προτάσεις / hints για το πώς θα πραγματοποιήσει την επίθεση ή ακόμη και να δει μια ολοκληρωμένη λύση με περιγραφή βήμα προς βήμα. Υπάρχουν μαθήματα στα οποία για να πραγματοποιηθεί η επίθεση χρειάζεται να χρησιμοποιήσουμε άλλα εργαλεία (πχ WebScarab ή Zed Attack Proxy κ.τ.λ.). Τα μαθήματα είναι όλων των βαθμών δυσκολίας, από εύκολα μέχρι και πιο δύσκολα και περίπλοκα.