Πανεπιστήμιο Πειραιώς – Τμήμα Πληροφορικής

Πρόγραμμα Μεταπτυχιακών Σπουδών

«Πληροφορική»

Μεταπτυχιακή Διατριβή

Τίτλος Διατριβής	**Ασφαλείς διαδικτυακές εφαρμογές.** **Επιθέσεις, Αδυναμίες, Αντίμετρα** Secure web Applications. Attacks, Vulnerabilities, Countermeasures
Ονοματεπώνυμο Φοιτητή	**Κωνσταντίνος Μαμαρέλης**
Πατρώνυμο	**Νικόλαος**
Αριθμός Μητρώου	**ΜΠΠΛ/ 1101**7
Επιβλέπων	**Νινέτα Πολέμη, Καθηγήτρια**

Ημερομηνία Παράδοσης **Ιούνιος** 2015

Τριμελής Εξεταστική Επιτροπή

(υπογραφή)	(υπογραφή)	(υπογραφή)
Όνομα Επώνυμο Βαθμίδα	Όνομα Επώνυμο Βαθμίδα	Όνομα Επώνυμο Βαθμίδα

Περιεχόμενα

Ευχαριστίες

Ολοκληρώνοντας τη μεταπτυχιακή μου διατριβή θα ήθελα να ευχαριστήσω την κ. Νινέτα Πολέμη και τον κ. Θάνο Καραντζιά για την καθοδήγηση τους. Επιπλέον, θα ήθελα να ευχαριστήσω θερμά όλους τους φίλους και συνεργάτες για τη βοήθεια και τη κατανόηση που έδειξαν καθ' όλη τη διάρκεια των σπουδών μου.

Περίληψη

Το αντικείμενο της παρούσας μεταπτυχιακή εργασίας είναι η ανάπτυξη μιας ασφαλούς διαδικτυακής εφαρμογής, η οποία στο μεγαλύτερο μέρος της δεν βασίζεται σε κάποια μεθοδολογία ανάπτυξης ή κάποιο σύστημα διαχείρισης περιεχομένου. Για το λόγο αυτό, πραγματοποιήθηκε διεξοδική έρευνα σχετικά με την σωστή εφαρμογή όλων των **μηχανισμών ασφαλείας** που μπορούν να χρησιμοποιηθούν για την προστασία από διάφορες **επιθέσεις** καθώς επίσης και για την εξάλειψη των **αδυναμιών** που μπορεί να υπάρχουν σε μια εφαρμογή.

Η δομή της εργασίας έχει ως ακολούθως:

- Στο **πρώτο κεφάλαιο** γίνεται μια σύντομη αναφορά σε βασικές έννοιες που σχετίζονται με θέματα ασφαλείας.

- Στο **δεύτερο κεφάλαιο** αναφέρονται με κάθε λεπτομέρεια όλα όσα θα πρέπει να γνωρίζει ένας προγραμματιστής διαδικτυακών εφαρμογών σχετικά με τις αδυναμίες, τις επιθέσεις και τα μέτρα προστασίας που αφορούν τις διαδικτυακές εφαρμογές.

- Στο **τρίτο κεφάλαιο** παρουσιάζονται όλες οι λειτουργίες που παρέχει η εφαρμογή στους χρήστες της καθώς επίσης και οι μηχανισμοί ασφαλείας που χρησιμοποιούνται.

- Το **τέταρτο κεφάλαιο** περιλαμβάνει τα συμπεράσματα που προέκυψαν από την έρευνα που πραγματοποιήθηκε.

- Στο **πέμπτο κεφάλαιο** αναγράφονται όλες οι μελλοντικές επεκτάσεις που μπορούν να υλοποιηθούν σε βάθος χρόνου.

Abstract

This dissertation focuses on the development of a secure web application which is not mainly based on a specific framework or a Content Management System. Therefore, a thorough research has been conducted for the proper implementation of **countermeasures** which can be used to protect against various **attacks** and also to eliminate the **weaknesses** that may exist in an application.

The structure of this dissertation is as follows:

- The **first chapter** is a quick reference to basic concepts related to security issues.

- The **second chapter** presents in detail everything a web developer needs to know on weaknesses, attacks and countermeasures which are related to web applications.

- In the **third chapter** there are presented all functionalities which are provided from the application to the users as well as all those countermeasures used.

- The **fourth chapter** contains the conclusions of the conducted research.

- The **fifth chapter** lists all future extensions that can be implemented over time.

Εισαγωγή-Κεφάλαιο 1- Βασικές έννοιες

1.1 Το διαδίκτυο

Η λέξη Διαδίκτυο προέρχεται από τις λέξεις Διασύνδεση Δικτύων και αναφέρεται σε ένα σύνολο υπολογιστών και δικτύων που συνδέονται μεταξύ τους σε ένα παγκόσμιο δίκτυο έτσι ώστε να μπορούν να επικοινωνούν και να μοιράζονται πληροφορίες. Η επικοινωνία μεταξύ των διασυνδεδεμένων υπολογιστών βασίζεται στην ανταλλαγή μηνυμάτων (**πακέτων**) με τη χρήση διάφορων **πρωτοκόλλων** (τυποποιημένοι κανόνες επικοινωνίας), τα οποία υλοποιούνται σε επίπεδο υλικού και λογισμικού. Το διαδίκτυο, είναι γνωστό και με την αγγλική άκλιτη ονομασία Internet η οποία προέρχεται από τις λέξεις International Network που σημαίνει Διεθνές Δίκτυο Υπολογιστών. [1]

Το διαδίκτυο κατά τα πρώτα χρόνια της δημιουργίας του ήταν ένα κλειστό δίκτυο που αποτελούνταν από συγκεκριμένους υπολογιστές που βρίσκονταν σε γνωστούς και συγκεκριμένους τόπους μιας κλειστής κοινότητας. Όταν όμως αυτό το δίκτυο έγινε ένα παγκόσμιο ανοιχτό δίκτυο υπολογιστών (παίρνοντας την μορφή που έχει σήμερα), άρχισαν να δημιουργούνται προβλήματα από άτομα με κακόβουλες προθέσεις. [2]

1.2 Τι είναι μια εφαρμογή διαδικτύου

Διαδικτυακή εφαρμογή (web application ή web app) ονομάζεται μια εφαρμογή η οποία είναι διαθέσιμη στους χρήστες της μέσω του Διαδικτύου (Internet) ή μέσω ενός ενδοδικτύου (Intranet). Οι χρήστες μπορούν να αλληλεπιδρούν με αυτή χρησιμοποιώντας, ως πελάτη (client), συνήθως κάποιον περιηγητή (browser).

Οι εφαρμογές αυτές βρίσκονται σε διακομιστές / εξυπηρετητές (servers) και ένα από τα πολλά πλεονεκτήματα που έχουν είναι ότι επιτρέπουν την πρόσβαση σε περισσότερους του ενός χρήστη ανεξάρτητα από την συσκευή που διαθέτουν και την τοποθεσία στην οποία βρίσκονται.

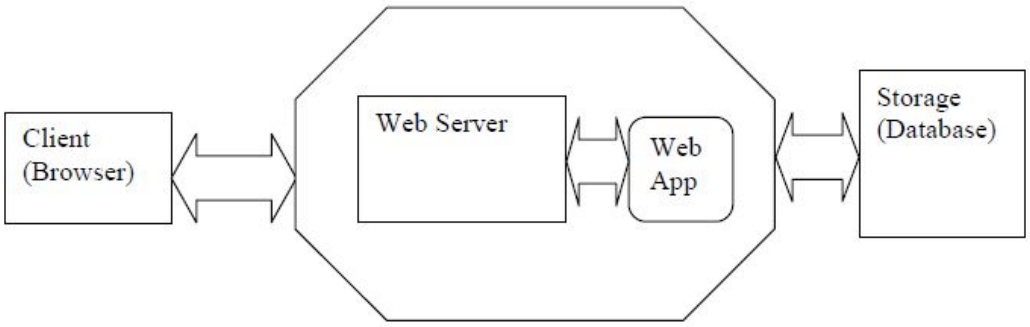
Από την πλευρά του προγραμματιστή (developer) το βασικότερο πλεονέκτημα είναι ότι δεν χρειάζεται να φτιάξει ξεχωριστό client για τους διάφορους τύπους ηλεκτρονικών υπολογιστών οι οποίοι ενδεχομένως να έχουν εγκατεστημένο διαφορετικό λειτουργικό σύστημα (Unix, Windows), αφού οι εν λόγω εφαρμογές εκτελούνται μέσω περιηγητή (browser). Επιπλέον, δεν παίζει κανέναν ρόλο αν το λειτουργικό σύστημα του εξυπηρετητή είναι το ίδιο με αυτό που έχει εγκατεστημένο ο χρήστης στον υπολογιστή του, όπως επίσης δεν παίζει κανένα ρόλο αν για παράδειγμα η εφαρμογή έχει αναπτυχθεί σε Java και ο περιηγητής σε C++. Με απλά λόγια, υποστηρίζεται η διαλειτουργικότητα. [3]

Οι διαδικτυακές εφαρμογές συνήθως αποτελούνται από λογικά μέρη που λέγονται "σειρές" (tiers). Σε αντίθεση με τις παραδοσιακές εφαρμογές, που αποτελούνται από μια σειρά, οι διαδικτυακές αποτελούνται από τουλάχιστον τρεις (n-tiered). Στην απλή μορφή της δομής οι τρεις αυτές σειρές είναι οι εξής:

1η Η παρουσίαση (Presentation)

2η Η εφαρμογή (Application) ή Business logic και

3η Η αποθήκευση (storage)



Εικόνα 1.0.1 - **Αρχιτεκτονική τριών σειρών** [5]

Η παρουσίαση γίνεται μέσω ενός περιηγητή, ο οποίος δίνει στο χρήστη την δυνατότητα πρόσβασης και αλληλεπίδρασης με την εφαρμογή μέσα από το γραφικό περιβάλλον της (Graphical User Interface).

Η δεύτερη σειρά αποτελείται από ένα ή περισσότερα αρχεία τα οποία βρίσκονται αποθηκευμένα μέσα σε φακέλους στον περιηγητή. Τα αρχεία αυτά μπορεί να είναι π.χ. εικόνες, video κα., αλλά και αρχεία που περιέχουν κώδικα (scripts). Σε αυτό το σημείο θα πρέπει να αναφερθεί ότι για να αναπτυχθεί μια διαδικτυακή εφαρμογή συνδυάζονται server-side-scripts (π.χ. PHP, ASP κλπ) με client-side-scripts (π.χ. JavaScript, html). Ένα server-side-script καθορίζει τις βασικές λειτουργίες που γίνονται στον εξυπηρετητή (π.χ. αποθήκευση, ανάκτηση πληροφοριών) ενώ ένα client-side-script ευθύνεται για τον τρόπο με τον οποίο παρουσιάζονται οι πληροφορίες στον περιηγητή του χρήστη (client) και με τη γενικότερη μορφή (interface).

Όσον αφορά το 3 μέρος, συνήθως πρόκειται για μια βάση δεδομένων όπου και αποθηκεύονται τα δεδομένα.

Με βάση τα παραπάνω καταλήγουμε στο ότι ο περιηγητής (client) στέλνει αιτήματα (requests) στην εφαρμογή (με την μορφή πακέτων), η οποία με την σειρά της "απαντάει" / "εξυπηρετεί" τον πελάτη (client) υποβάλλοντας στην Βάση Δεδομένων "ερωτήματα" (queries), "ενημερώσεις" (updates) κα. [3] [4] [5] [6]

1.3 **Ασφάλεια** (Security)

Ο αγγλικός όρος "security", φέρεται να είναι λατινικής προέλευσης, αφού προέρχεται από τις αντίστοιχες λατινικές λέξεις "se" που σημαίνει "χωρίς" και "cura" που σημαίνει "φροντίδα". Δηλαδή, η έννοια της ασφάλειας σε ένα σύστημα μπορεί και να θεωρηθεί ως μια επιθυμητή ιδιότητα - κατάσταση, κατά την οποία οι χρήστες του απαλλάσσονται κάθε έγνοιας και φροντίδας, ως προς τη σωστή λειτουργία του. Παρόλο που ο όρος ασφάλεια φαίνεται να έχει μια προφανή σημασία, το νόημα του όρου είναι πιο σύνθετο σε ότι αφορά την επιστήμη της πληροφορικής, τόσο σε επίπεδο hardware όσο και σε επίπεδο software. [7]

Στην επιστήμη της πληροφορικής, ασφάλεια είναι εκείνος ο κλάδος που ασχολείται με την προστασία όλων των αγαθών (assets) που απαρτίζουν μια εφαρμογή είτε από συμπτωματική είτε από κακόβουλη ενέργεια. Έχει σαν βασικές απαιτήσεις την διασφάλιση της ακεραιότητας (Integrity), της αυθεντικότητας (Authentication), της εγκυρότητας (Validation), της εμπιστευτικότητας (Confidentiality), της διαθεσιμότητας (Availability) και της μη αποποίησης ευθύνης (Non repudiation). [8]

Ακεραιότητα (Integrity): Και μόνο η ετυμολογική ανάλυση της λέξης αρκεί για να καταλάβει κάποιος το νόημα της. Πιο συγκεκριμένα, ο έλεγχος της ακεραιότητας εστιάζει στην εσφαλμένη τροποποίηση (π.χ. εγγραφή, ανανέωση ή διαγραφή) των πληροφοριών που ενδεχομένως μπορεί να πραγματοποιηθεί με ή χωρίς δόλο. Μια εφαρμογή πρέπει να επιτρέπει τέτοιες ενέργειες μόνο σε εξουσιοδοτημένους χρήστες και παράλληλα να εξασφαλίζει την ακρίβεια και την πληρότητα των διακινούμενων πληροφοριών, ώστε οι πληροφορίες που αποστέλλονται να καταλήγουν στον τελικό τους αποδέκτη όπως ακριβώς στάλθηκαν.

Αυθεντικότητα (Authentication): Η αυθεντικότητα είναι η διαδικασία που αποσκοπεί στην εξακρίβωση της ταυτότητας ενός χρήστη. Συνεπώς, από την στιγμή που η ταυτότητα του χρήστη πιστοποιείται, το σύστημα του δίνει και τα ανάλογα δικαιώματα (permissions). Ο έλεγχος αυθεντικότητας παρέχεται συνήθως μέσω ψηφιακών υπογραφών.

Εγκυρότητα (Validation): Η εγκυρότητα εξασφαλίζει ότι τα δεδομένα είναι ακριβή και πλήρη.

Εμπιστευτικότητα (Confidentiality): Είναι έννοια στενά συνδεδεμένη με την ιδιωτικότητα (privacy) και τη μυστικότητα (secrecy). Αφορά την προστασία των ευαίσθητων πληροφοριών από άτομα που δεν είναι εξουσιοδοτημένα από το σύστημα ή είναι εξουσιοδοτημένα με περιορισμένα δικαιώματα (restricted permissions). Η εμπιστευτικότητα συνήθως εξασφαλίζεται με τεχνικές κρυπτογράφησης.

Διαθεσιμότητα (Availability): Είναι εκείνη η υπηρεσία ασφαλείας που επιτρέπει στους εξουσιοδοτημένους χρήστες να χρησιμοποιούν την εφαρμογή όποτε θέλουν, δίχως αδικαιολόγητη καθυστέρηση και σύμφωνα πάντα με τις προδιαγραφές του συστήματος.

Μη αποποίηση ευθύνης (Non repudiation): Μη αποποίηση ευθύνης σημαίνει ότι ένας χρήστης δεν μπορεί να αρνηθεί την εκτέλεση μιας λειτουργίας, και κανένα από τα συναλλασσόμενα μέρη δεν έχει τη δυνατότητα να αρνηθεί τη συμμετοχή του σε μια συναλλαγή. Οι υπηρεσίες μη αποποίησης ευθύνης πρέπει, σε περίπτωση που χρειαστεί, να μπορούν να αποδείξουν την προέλευση, μεταφορά και παραλαβή των δεδομένων. Σε τέτοιες περιπτώσεις συνηθίζεται η καταγραφή των ενεργειών των εξουσιοδοτημένων χρηστών **(Security auditing)**. [10]

Για τη καλύτερη κατανόηση της έννοιας της ασφάλειας θα πρέπει να αναλύσουμε τις τρεις συνεχείς και διαφορετικές μεταξύ τους δράσεις που αυτή απαιτεί:

Η **πρόληψη (prevention)** αποσκοπεί στην λήψη μέτρων που μας επιτρέπουν να προλαβαίνουμε τη δημιουργία επικίνδυνων καταστάσεων. Αποτελεί την ουσία της ασφάλειας, καθώς χρησιμεύει ως μονάδα ποσοτικής μέτρησης έναντι της ανίχνευσης και αντίδρασης.

Ανίχνευση (detection) είναι η λήψη μέτρων που μας επιτρέπουν να αντιληφθούμε πως, πότε και από ποιόν έχει προκληθεί κάποια ζημιά. Συνεπικουρεί στην ανίχνευση τυχόν κενών και προβλημάτων ασφάλειας, μόλις τα προληπτικά μέτρα τεθούν σε εφαρμογή.

Αντίδραση (reaction) είναι η λήψη μέτρων που μας επιτρέπουν να αποκαταστήσουμε τις ζημιές που έχουν προκληθεί χρησιμοποιώντας τους κατάλληλους μηχανισμούς. [8] [9] [5]

1.4 Βασικές έννοιες

Ως **αγαθό (asset)** ορίζεται κάθε αντικείμενο ή πόρος το οποίο πρέπει να προστατευτεί.

Τα αγαθά χωρίζονται στις εξής κατηγορίες:

1. Φυσικά αγαθά (Physical Assets): όπως υπολογιστές, δικτυακή υποδομή κ.α.
2. Αγαθά Δεδομένων (Data Assets): όπως αρχεία (ηλεκτρονικά, έντυπα) κ.α.
3. Αγαθά Λογισμικού (Software Assets): Τα λειτουργικά συστήματα και το λογισμικό εφαρμογών.

Ως **συνέπεια (impact)** ή αντίκτυπος ή επίπτωση ορίζεται η απώλεια που μπορεί να προκληθεί από την τροποποίηση ενός αγαθού (asset). Οι συνέπειες διακρίνονται σε άμεσες και έμμεσες.

- Στις άμεσες συνέπειες συμπεριλαμβάνονται το κόστος επαναγοράς και διαμόρφωσης και άλλα οικονομικά θέματα.
- Στις έμμεσες συνέπειες συμπεριλαμβάνονται νομικές, κοινωνικές και άλλες επιπτώσεις που μπορούν να βλάψουν την αξιοπιστία και το κύρος.

Απειλή (threat) είναι οποιοδήποτε πιθανό περιστατικό, κακόβουλο ή όχι, που θα μπορούσε να παραβιάσει την ασφάλεια μιας εφαρμογής ή ενός συστήματος γενικότερα και να προκαλέσει ζημιά υπό μορφή καταστροφής, κοινοποίησης, τροποποίησης των αγαθών (assets) ή και άρνηση της υπηρεσίας (Denial Of Service).

Ευπάθεια (vulnerability) ονομάζεται μια "τρύπα" ή μια **αδυναμία (weakness)** της εφαρμογής, η οποία μπορεί να οφείλεται σε ένα τρωτό σημείο κατά την σχεδίασή της ή σε ένα σφάλμα κατά την υλοποίησή της και μπορεί να επιτρέψει την υλοποίηση μιας απειλής (threat). Ο όρος της ευπάθειας πολύ συχνά χρησιμοποιείται εσφαλμένα. Οι ευπάθειες ενός συστήματος χωρίζονται σε σφάλματα (bugs) και ελαττώματα (flaws).

- Τα σφάλματα είναι ευπάθειες που εντοπίζονται αποκλειστικά στον κώδικα, ενώ
- Τα ελαττώματα είναι ευπάθειες που είναι πιο εγγενείς, καθώς προέρχονται από κακή σχεδίαση.

Σύμφωνα με την Microsoft, περίπου το 50% των ευπαθειών είναι ελαττώματα. Η ευπάθεια πρέπει να διακρίνεται από τους όρους: απειλή (threat), επίθεση (attack) και αντίμετρα (countermeasures) διότι:

1. Οι απειλές είναι αναγκαίο να εκμεταλλευτούν συγκεκριμένες ευπάθειες, προκειμένου να προκαλέσουν ένα περιστατικό ασφάλειας και
2. Οι απειλές, οι ευπάθειες, και οι επιδράσεις τους πρέπει να συνδυαστούν ώστε να παράσχουν μια μονάδα μέτρησης του κινδύνου.

Εκμετάλλευση (exploit) ή αλλιώς "αδυναμίες - κενά - τρύπες". Πρόκειται για προγραμματιστικά σφάλματα που εντοπίζονται στο λογισμικό και έχουν ως αποτέλεσμα την μη εξουσιοδοτημένη πρόσβαση στο σύστημα ή/και την κατάρρευση από άρνηση παροχής υπηρεσίας (Denial Of Service) του δικτύου του συστήματος. Οι εκμεταλλεύσεις προσδιορίζουν έναν προκαθορισμένο τρόπο παραβίασης της ασφάλειας ενός συστήματος σε μια ευπάθεια. Κατηγοριοποιούνται στις εξής κατηγορίες:

- ✓ Σε εκείνες που εκμεταλλεύονται αδυναμίες του συστήματος με σκοπό την πρόσβαση σε αυτό και

- ✓ Σε αυτές που στοχεύουν στην απόκτηση περισσότερων δικαιωμάτων (permissions) μέσα στο σύστημα.

Οι αναλυτές χρησιμοποιούν κατάλληλα εργαλεία για να εντοπίσουν "κενά" που βασίζονται στη λανθασμένη ή μη παραμετροποίηση του λειτουργικού συστήματος, του δικτύου και των προ-εγκατεστημένων εφαρμογών που διαθέτει το εκάστοτε σύστημα. Η εύρεση τέτοιων "κενών" ασφαλείας και η διόρθωση τους, μπορούν να εμποδίσουν ενδεχόμενες επιθέσεις τόσο σε επίπεδο συστήματος όσο και σε επίπεδο δικτύου.

Επίθεση (Attack) είναι μια ενέργεια που εκμεταλλεύεται τις ευπάθειες και υλοποιεί μια απειλή. Η διαφορά μιας απειλής και μιας επίθεσης είναι ότι στην πρώτη περίπτωση μιλάμε για ένα πιθανό γεγονός και στην δεύτερη για μια επιτυχή εκμετάλλευση μιας ευπάθειας του συστήματος.

Αντίμετρα ή μέτρα προστασίας (Countermeasures) είναι ο μηχανισμός ή η διαδικασία εκείνη (τεχνολογίες ή μοντέλα άμυνας) που αποσκοπεί στην μείωση ή την αποτροπή των επιμέρους κινδύνων στους οποίους εκτίθεται ένα αγαθό (asset). Τα αναγκαία αντίμετρα σε μια εφαρμογή πρέπει να αναγνωριστούν με την χρήση της ανάλυσης κινδύνου (risk analysis) έτσι ώστε να διασφαλιστεί ότι η εφαρμογή προστατεύεται από κοινούς τύπους επιθέσεων. Οποιαδήποτε αδυναμία ή ρωγμή στη σχεδίαση των αντιμέτρων ή ακόμα και η παράλειψη ενός συγκεκριμένου αντιμέτρου μπορεί να έχει ως αποτέλεσμα μια ευπάθεια, η οποία μπορεί να είναι ικανή να καταστήσει την εφαρμογή ευάλωτη σε επιθέσεις. [5]

Ωστόσο, πολλές φορές η παράληψη συγκεκριμένων αντιμέτρων γίνεται σκόπιμα. Το γεγονός αυτό οφείλεται συνήθως σε οικονομικούς και λειτουργικούς λόγους. Συνήθως, η εφαρμογή ενός επιπρόσθετου μέτρου προστασίας που ενσωματώνεται σε μια εφαρμογή ή ένα σύστημα αυξάνει το κόστος συντήρησης και ταυτόχρονα την καθιστά δύσχρηστη.

Σε αυτό το σημείο θα πρέπει να τονιστεί ότι ακόμα και αν η εφαρμογή συνδυάζει τον μέγιστο εφικτό αριθμό αντιμέτρων δεν μπορεί να προστατέψει τον χρήστη της αν και ο ίδιος δεν γνωρίζει και δε λαμβάνει μέτρα από την δική του πλευρά. Πολλοί είναι οι χρήστες που πέφτουν θύματα "κατασκοπίας" επειδή δεν λαμβάνουν τα απαραίτητα μέτρα στον υπολογιστή τους. [7] [9] [10]

1.5 Πολιτική Ασφάλειας

Η πολιτική ασφάλειας (security policy) περιλαμβάνει ένα σύνολο κανόνων, αρμοδιοτήτων, ευθυνών και καθηκόντων που προσδιορίζουν επακριβώς το ρόλο κάθε εμπλεκόμενου μέσα σε ένα σύστημα, μια εφαρμογή κ.τ.λ. Με το καθορισμό της πολιτικής ασφαλείας υπαγορεύονται μέτρα προστασίας τα οποία υλοποιούνται με μηχανισμούς ασφάλειας, διοικητικά μέτρα αλλά και με ηθικό-κοινωνικές αντιλήψεις, αρχές και παραδοχές, έχοντας σαν στόχο την προφύλαξη από κάθε είδους απειλή που μπορεί να είναι τυχαία ή σκόπιμη. [7] [9] [4] [10]

Τα είδη των πολιτικών ασφάλειας είναι δυο:

- ✓ οι **τεχνικές (computer oriented)**, οι οποίες περιλαμβάνουν Πολιτικές Ασφάλειας Πληροφοριών, Πολιτικές Ασφάλειας Λειτουργικών Συστημάτων και Πολιτικές Ασφάλειας Δικτύων Υπολογιστών και
- ✓ οι **οργανωτικές (human oriented)**, όπου διαμορφώνονται Πολιτικές Ασφάλειας Πληροφοριακών Συστημάτων, οι οποίες άπτονται όλων των πτυχών των πληροφοριακών συστημάτων.

Από την άλλη πλευρά, οι πολιτικές ασφάλειας μπορούν να έχουν τις εξής μορφές:

- ✓ **Ατομικές (Individual Security Policies)**. Οι πολιτικές αυτές αναπτύσσονται ανά σύστημα ή εφαρμογή (π.χ. Πολιτική ασφάλειας για τη χρήση του email). Παρέχουν αποσπασματική διαχείριση της ασφάλειας του συστήματος και έχουν μεγάλη πολυπλοκότητα στη συντήρησή τους, είναι ωστόσο αποτελεσματικές όταν υπάρχουν αυτόνομες εφαρμογές και υπολογιστικά συστήματα που δε συνδέονται μεταξύ τους.
- ✓ **Αναλυτικές (Comprehensive Security Policies)**, όπου υπάρχει ένα ενιαίο έγγραφο που αναφέρεται σε όλα τα υπολογιστικά συστήματα, τις εφαρμογές και τις διαδικασίες του. Είναι μεγάλες σε όγκο, και κατά συνέπεια όχι πολύ ευχάριστες, ενώ οι οδηγίες και διαδικασίες που περιλαμβάνονται είναι σε γενικό επίπεδο, χωρίς λεπτομέρειες.
- ✓ **Αρθρωτές (Modular Security Policies)**. Σε αυτές υπάρχει ένα ενιαίο έγγραφο με παραρτήματα που περιγράφουν τις επιμέρους πολιτικές. Το έγγραφο αυτό μπορεί να είναι σε μορφή υπερκειμένου (hypertext).

1.6 Εμπλεκόμενα πρόσωπα με την ασφάλεια

Τα πρόσωπα που ασχολούνται με την ασφάλεια χωρίζονται σε κατηγορίες με βάση τα κίνητρα και τις προθέσεις τους. Το μεγαλύτερο ποσοστό αυτών των ανθρώπων αποσκοπεί στην εξοικονόμηση χρηματικών ποσών με πολλούς έμμεσους και άμεσους τρόπους. Ωστόσο, υπάρχει και ένα σεβαστό ποσοστό ανθρώπων, οι οποίοι πραγματοποιούν επιθέσεις είτε για να αποδείξουν ότι απλά έχουν την γνώση και μπορούν να το κάνουν είτε για ιδεολογικούς λόγους. Παραδείγματα τέτοιου είδους επιθέσεων στην Ελλάδα είχαμε το πρώτο τρίμηνο του 2012, όπου η γνωστή ομάδα "Anonymous" επιτέθηκε σε αρκετές κυβερνητικές ιστοσελίδες και κατάφερε να τις θέσει εκτός λειτουργίας, κυρίως με **επιθέσεις άρνησης υπηρεσίας (Denial Of Service)**. Οι σημαντικότερες κατηγορίες είναι οι εξής: [9] [5]

Hackers: είναι τα άτομα εκείνα που εισβάλουν χωρίς εξουσιοδότηση σε υπολογιστές και δίκτυα. Ο όρος hacker απέκτησε αρνητική σημασία αρκετά χρόνια μετά την εμφάνισή του. Αρχικά, ως hacker χαρακτηρίζονταν ο ευρηματικός και χαρισματικός προγραμματιστής. Στους κύκλους των hacker ο όρος που χρησιμοποιείτε για τους εγκληματίες είναι **cracker**. Οι προθέσεις τους συχνά δεν είναι εχθρικές, ωστόσο σε κάθε περίπτωση, το hacking αποτελεί κακούργημα στις περισσότερες χώρες και τιμωρείτε με βαριές ποινές.

Ethical Hackers: Ένας "ηθικός" hacker αμείβεται για να προσπαθήσει να εισβάλει σε ένα σύστημα με σκοπό να αξιολογήσει το επίπεδο ασφαλείας του. Πολλές φορές είναι γνωστοί ως αξιολογητές ασφαλείας (Security testers).

Ερευνητές (Researchers): Ένας ερευνητής μπορεί να εργαστεί πολύ σκληρά στην προσπάθεια του να ανακαλύψει αδυναμίες σε πρωτόκολλα ασφάλειας και στη συνέχεια εκδίδει τα αποτελέσματά του στο διαδίκτυο. Συνήθως, είναι μέλη μη κερδοσκοπικών οργανισμών ή άλλων ανοιχτών κοινοτήτων.

Εγκληματίες (Criminals): Είναι τα άτομα που έχουν σαν στόχο την απόκτηση χρημάτων εξαπατώντας χρήστες κλέβοντας τα προσωπικά τους δεδομένα. Συνήθως πραγματοποιούν επιθέσεις προκειμένου να κλέψουν τον αριθμό της πιστωτικής κάρτας ενός χρήστη ή άλλα ευαίσθητα προσωπικά δεδομένα.

Ανταγωνιστές (Competitors): Ένας ανταγωνιστής δεν αποσκοπεί στην άμεση εξοικονόμηση χρημάτων αλλά στο να δυσφημήσει την αξιοπιστία και το επίπεδο ασφάλειας των επιχειρηματικών του αντιπάλων.

Εσωτερικοί εχθροί: Σε αυτή την κατηγορία κατατάσσονται χρήστες που έχουν εξ ορισμού πρόσβαση σε ευαίσθητα συστήματα και πληροφορίες. Κλασική περίπτωση τέτοιων χρηστών είναι οι δυσαρεστημένοι ή άπληστοι υπάλληλοι και αποτελούν την πιο σοβαρή απειλή για την ασφάλεια του συστήματος αφού ενδέχεται να γνωρίζουν και τα τρωτά σημεία της. [10]

1.7 Βασικές μέθοδοι ελέγχου ασφάλειας των εφαρμογών ιστού

Από την στιγμή που ένα σύστημα συνδέεται με το διαδίκτυο δεν μπορεί να υποστηρίξει κάνεις ότι είναι ασφαλές αλλά ούτε και το αντίθετο. Το βασικότερο χαρακτηριστικό της ασφάλειας είναι ότι αποτελεί αναπόσπαστο μέρος του σχεδιασμού ενός συστήματος και δεν μπορεί να είναι μια στατική διαδικασία η οποία εφαρμόζεται μόνο στην αρχή κατά την σχεδίασή του. Με την ραγδαία βελτίωση των τεχνολογικών εφαρμογών έπεται παράλληλα και η βελτίωση των εργαλείων που χρησιμοποιούν και οι κακόβουλοι χρήστες. Συνεπώς, η μόνη λύση είναι να πραγματοποιούνται συνεχείς έλεγχοι από κάποιον **"ethical" hacker**. Οι βασικές μέθοδοι ασφάλειας ελέγχου των διαδικτυακών εφαρμογών είναι η ακόλουθες:

Επιθεώρηση ασφάλειας (security audit): είναι μια διαδικασία κατά την οποία ένα σύστημα ελέγχεται με βάση ένα σύνολο από λίστες ελέγχου (checklists), οι οποίες περιλαμβάνουν έναν συνδυασμό διεθνή προτύπων ασφάλειας και πολιτικών του οργανισμού που χρησιμοποιεί το σύστημα. Προκειμένου ο ελεγκτής να καταλήξει σε ασφαλή συμπεράσματα πραγματοποιεί προσωπικές συνεντεύξεις, ανιχνεύσεις αδυναμιών, εξετάσεις των ρυθμίσεων, αναλύσεις των διαμοιρασμένων πόρων δικτύου και μελέτες των αρχείων καταγραφής (log files).

Αυτοαξιολόγηση ασφάλειας (security self-assement): Σε αυτήν τη μέθοδο δεν χρησιμοποιούνται συγκεκριμένα πρότυπα (standards) ή λίστες ελέγχου (checklists) για την αξιολόγηση του συστήματος. Ο στόχος προσδιορίζεται από την περιοχή που χρειάζεται διερεύνηση και βελτίωση στη θωράκισή της. Πρόκειται ουσιαστικά για έναν πιο λεπτομερή έλεγχο για τον εντοπισμό αδυναμιών που δίνει την δυνατότητα να οριστούν επίπεδα προτεραιότητας σε κάθε **αγαθό (asset)** που αξιολογείται. Αφού ολοκληρωθεί ο έλεγχος, δίνεται μια σειρά προτεραιότητας σε ότι αφορά την επιδιόρθωση των ευπαθειών που βρέθηκαν.

Άξιο αναφοράς είναι ότι οι δύο παραπάνω μέθοδοι απαιτούν την φυσική παρουσία μιας μεγάλης ομάδας ειδικών ασφαλείας στον τόπο που λειτουργεί ο οργανισμός, του οποίου η ασφάλεια της εφαρμογής ελέγχεται. Η ομάδα αυτή πρέπει να διαθέτει τις κατάλληλες λίστες ελέγχου, να έχει υψηλή τεχνογνωσία και να είναι άρτια συντονισμένη. Οι έλεγχοι τέτοιου είδους είναι εξαιρετικά χρονοβόροι, λόγω του ότι πρέπει να ολοκληρωθούν οι συνεντεύξεις, οι επιθεωρήσεις, οι αξιολογήσεις και οι έρευνες στην διάρκεια των οποίων αποκαλύπτεται και διαταράσσεται η λειτουργία του οργανισμού.

Δοκιμή διείσδυσης (penetration testing ή ethical hacking): Είναι μια μέθοδος που χρησιμοποιείται από ethical hackers και βοηθάει στο να κατανοήσει καλύτερα ο αμυνόμενος τα τρωτά σημεία του συστήματός του. Αποτελεί μια προσομοίωση επίθεσης (attack simulation), η οποία είναι ελεγχόμενη και πραγματοποιείται με εργαλεία και τεχνικές που χρησιμοποιούνται από τους hackers. Σκοπός της εν λόγω μεθόδου δεν είναι να εντοπίσει όλες τις ευπάθειες, αλλά να αποδείξει ότι η ασφάλεια του συστήματος μπορεί να διακυβευτεί από έναν εξωτερικό χρήστη. Η δοκιμή διείσδυσης, γνωστή και ως εσωτερική επιθεώρηση ασφάλειας (internal security auditing), μπορεί να πραγματοποιηθεί στη βάση μηδενικής γνώσης (zero knowledge) ή με πλήρη γνώση (full knowledge) του συστήματος, που δοκιμάζεται.

Σε αντίθεση με τις άλλες δυο μεθόδους, η δοκιμή διείσδυσης είναι πιο οικονομική, απαιτεί ελάχιστο προσωπικό χωρίς να είναι αναγκαία η φυσική παρουσία, παρέχει την δυνατότητα πλήρους αυτοματοποίησης, διαρκεί ελάχιστο χρόνο και μπορεί να επαναλαμβάνεται. Επιπλέον, δεν απαιτεί βαθιά γνώση της ελεγχόμενης εφαρμογής και δεν διαταράσσει τη λειτουργία της εφαρμογής ή του οργανισμού. Οι τεχνικές δοκιμής διείσδυσης κατατάσσονται σε δυο κατηγορίες:

Αυτοματοποιημένη (automated): όπου γίνεται χρήση εργαλείων ώστε οι έλεγχοι να εκτελούνται αυτόματα.

Χειροκίνητη (manual): όπου ο έλεγχος πραγματοποιείται βήμα προς βήμα χωρίς να υπάρχουν αυτοματισμοί επανάληψης παρόμοιων βημάτων.

Μία διαφορετική κατηγοριοποίηση των τεχνικών δοκιμής διείσδυσης βασίζεται στο βαθμό γνώσης που ο επιτιθέμενος διαθέτει για το σύστημα-στόχο:

Black Box: είναι οι τεχνικές που χρησιμοποιεί ο επιτιθέμενος όταν δεν γνωρίζει τίποτα για το σύστημα, χρησιμοποιώντας το γραφικό περιβάλλον της εφαρμογής (GUI) προσποιούμενος τον απλό επισκέπτη.

White Box: είναι οι τεχνικές που χρησιμοποιεί ο επιτιθέμενος που διαθέτει μερική ή πλήρη γνώση του υπό-εξέταση συστήματος. Δηλαδή, γνωρίζει την αρχιτεκτονική του λογισμικού, τις βιβλιοθήκες τρίτων κατασκευαστών, το λειτουργικό σύστημα, τις βάσεις δεδομένων κλπ. Ο απώτερος στόχος είναι η απόπειρα δημοσίευσης των κρίσιμων πληροφοριών του συστήματος.

Gray box: είναι η προσομοίωση μιας συστηματικής επίθεσης από καλά προετοιμασμένους εξωτερικούς χρήστες, που έχουν περιορισμένη γνώση της υποδομής και των μηχανισμών άμυνας του συστήματος, ή από χρήστες που έχουν πρόσβαση με περιορισμένα δικαιώματα (permissions). [7] [9]

1.8 Οργανισμοί που ασχολούνται με την ασφάλεια

Με την εμφάνιση του Web 2.0, ο αυξημένος διαμοιρασμός πληροφοριών μέσα από την κοινωνική δικτύωση και την αύξηση της χρήσης του διαδικτύου, ως μέσο για την άσκηση επιχειρηματικής δραστηριότητας και παροχής υπηρεσιών πληροφοριών, είχε ως επακόλουθο οι ιστοσελίδες να δέχονται επιθέσεις απευθείας. Αυτό είχε ως αποτέλεσμα την ίδρυση ανοιχτών κοινοτήτων με στόχο την σωστή ενημέρωση για την πρόληψη και προστασία από κακόβουλους χρήστες. Λειτουργούν ως μη κερδοσκοπικοί οργανισμοί, ακολουθούν την ιδεολογία του ελεύθερου - ανοιχτού λογισμικού και παρέχουν δωρεάν, όλα τα πρότυπα, έγγραφα και εργαλεία που αναπτύσσουν. Οι πιο γνωστοί είναι οι εξής:

Open Web Application Security Project (OWASP): Το OWASP είναι μια ανοιχτή κοινότητα αφιερωμένη στην ενημέρωση των οργανισμών για το πώς μπορούν να αναπτύξουν, προμηθευτούν και συντηρήσουν ασφαλείς εφαρμογές. Για το λόγο αυτό, διοργανώνει διεθνή συνέδρια για την ενημέρωση των επαγγελματιών του χώρου. Έχει έδρα την Αμερική και συνεργάζεται με προγραμματιστές από πάρα πολλές χώρες, συμπεριλαμβανομένης και της Ελλάδας.

Είναι ελεύθερη από τις πιέσεις της αγοράς γιατί δεν σχετίζεται με καμία εταιρεία τεχνολογίας, γεγονός που επιτρέπει να παρέχει ανεπηρέαστες, πρακτικές και οικονομικές πληροφορίες για την ασφάλεια των εφαρμογών. Όπως και άλλα έργα ελεύθερου και ανοιχτού λογισμικού, έτσι και το OWASP παράγει υλικό με έναν ανοικτό και συνεργατικό τρόπο.

National Institute of Standards and Technology (NIST): Το Εθνικό Ινστιτούτο Προτύπων και Τεχνολογίας (NIST) ιδρύθηκε το 1901 και είναι πλέον μέρος του Υπουργείου Εμπορίου των ΗΠΑ. Το Εργαστήριο πληροφορικής του NIST, μεταξύ άλλων δραστηριοτήτων, αναλαμβάνει την ανάπτυξη και διαχείριση τεχνικών προτύπων.

Η συλλογή με τα πρότυπα της σειράς 800 (800-series), σχετίζεται με την διαχείριση της ασφάλειας των πληροφοριών. Ο οδηγός NIST SP 800-115 "Technical Guide to Information Security Testing and Assessment" ασχολείται με τον έλεγχο της ασφάλειας των δικτύων και αναφέρεται στις βασικές τεχνικές πτυχές σχετικά με την διαχείριση των αξιολογήσεων της ασφάλειας των πληροφοριών. Παρουσιάζει μεθόδους και τεχνικές ελέγχου που ένας οργανισμός μπορεί να ακολουθήσει προκειμένου να αξιολογήσει τα συστήματα και τα δίκτυα του. Πραγματοποιεί πρακτικές συστάσεις σχετικά με τον σχεδιασμό, την υλοποίηση των ελέγχων ασφαλείας και των διαδικασιών αξιολόγησης για την εύρεση **ευπαθειών (vulnerabilities)** σε ένα σύστημα ή ένα δίκτυο.

Web Application Security Consortium (WASC): Ο μη κερδοσκοπικός οργανισμός WASC απαρτίζεται από μια διεθνή ομάδα ειδικών που παράγουν πρότυπα ασφάλειας βέλτιστων πρακτικών για διαδικτυακές εφαρμογές. Απευθύνεται σε προγραμματιστές διαδικτυακών εφαρμογών, επαγγελματίες της ασφάλειας, προμηθευτές λογισμικού κα., παρέχοντάς τους πρόσβαση σε πληθώρα κατηγοριοποιημένων πληροφοριών.

Institute for Security & Open Methodologies (ISECOM): Ένα ακόμα μη κερδοσκοπικό ίδρυμα για την ασφάλεια και τις ανοιχτές μεθοδολογίες είναι το ISECOM. Παρέχει το εγχειρίδιο Open Source Security Testing Methodology (OSSTMM) για τον έλεγχο και την ανάλυση της ασφάλειας με χρήση μιας τυποποιημένης μεθοδολογίας. [7] [9]

2 ΚΕΦΑΛΑΙΟ-ΕΥΠΑΘΕΙΕΣ - ΕΠΙΘΕΣΕΙΣ - ΜΗΧΑΝΙΣΜΟΙ ΑΣΦΑΛΕΙΑΣ

Ένας προγραμματιστής διαδικτυακών εφαρμογών θα πρέπει να γνωρίζει ότι η ασφάλεια δεν είναι απλά μια λίστα από πράγματα που πρέπει να κάνει κάποιος. Η ασφάλεια είναι ένας τρόπος σκέψης, ένας τρόπος για να βλέπει κάνεις τα πράγματα τόσο από την πλευρά του αμυνόμενου όσο και από την πλευρά του επιτιθέμενου. Μεταξύ άλλων, η ενασχόληση με την ασφάλεια περιλαμβάνει:

- ✓ Το κατανόηση των βασικών αρχών και του λεξιλογίου(που αναφέρθηκαν στο 1ο κεφάλαιο) χωρίς πολλές λεπτομέρειες.
- ✓ Τον έλεγχο για **ευπάθειες (weaknesses)** από την πλευρά της εφαρμογής, του διακομιστή (server) και του δικτύου
- ✓ Την ορθή χρήση των κατάλληλων **αντιμέτρων (countermeasures)** για τη μείωση ή την αποτροπή των επιμέρους κινδύνων και **επιθέσεων (attacks)** από κακόβουλους χρήστες.

2.1 **Ευπάθειες** (Weaknesses)

2.1.1 **Κακή διαμόρφωση Εφαρμογής** (Application Misconfiguration)

Η ευπάθεια της κακής διαμόρφωσης μιας εφαρμογής οφείλεται σε διάφορες ρυθμίσεις και λειτουργίες που είναι ενεργοποιημένες από προεπιλογή για να κάνουν πιο εύκολη την εγκατάσταση (installation), την παραμετροποίηση (configuration) και τη συντήρηση αυτής. Ένας hacker μπορεί να εκμεταλλευτεί μεταξύ άλλων:

- Τους ειδικούς μηχανισμούς πρόσβασης
- Τις εναλλακτικές διόδους πρόσβασης (backdoors)
- Τα προεπιλεγμένα συνθηματικά (usernames & passwords) που αρχικά δίνονται με μη κρυπτογραφημένο κείμενο
- Τα εσφαλμένα δικαιώματα (permissions) αρχείων και φακέλων που είναι προσβάσιμα μέσω του εξυπηρετητή (server)
- Τις προεπιλεγμένες ρυθμίσεις ασφαλείας (οι οποίες συνήθως είναι στο χαμηλότερο επίπεδο)
- Την ενεργοποιημένη λειτουργία αποσφαλμάτωσης (debug)

Όλα τα παραπάνω, μπορεί να δώσουν σε έναν hacker την δυνατότητα να παρακάμψει τις μεθόδους ελέγχου ταυτότητας και να αποκτήσει πρόσβαση σε ευαίσθητες πληροφορίες, πολλές φορές με αυξημένα προνόμια.

Για να αποφύγουμε κάτι τέτοιο, το μόνο που πρέπει να κάνουμε είναι να αλλάξουμε όλες τις προεπιλεγμένες ρυθμίσεις καθώς και τα αρχικά συνθηματικά (usernames και passwords) μετά την εγκατάσταση της εφαρμογής.

Αντίστοιχη αναφορά υπάρχει και στο Top 10 του OWASP με τίτλο "Επισφαλείς ρυθμίσεις ασφαλείας" (Security Misconfiguration). [7] [11]

2.1.2 **Εμφάνιση λίστας περιεχομένων καταλόγων (**Directory Indexing)

Η εμφάνιση αυτόματης λίστας περιεχομένων καταλόγου είναι μία λειτουργία του εξυπηρετητή που ενεργοποιείται όταν κάποιος χρήστης βρεθεί σε μια τοποθεσία (directory) όπου το εξ ορισμού αρχείο (π.χ. index .html || .php || .asp κλπ.) δεν εντοπίζεται. Σε αυτή την περίπτωση, ο εξυπηρετητής λαμβάνει το αίτημα (request) και διαμορφώνει δυναμικά μια λίστα με όλα τα αρχεία που περιέχονται στο συγκεκριμένο κατάλογο (directory). Ουσιαστικά, η λειτουργία αυτή ισοδυναμεί με την εντολή "ls" σε Unix ή με την εντολή "dir" στα Windows με την διαφορά ότι η παρουσίαση των αποτελεσμάτων γίνεται σε μορφή HTML.

Η ευπάθεια αυτή, αν και δυνητικά αβλαβής, θα μπορούσε να δώσει σε έναν hacker τις απαραίτητες πληροφορίες για να δρομολογήσει περαιτέρω επιθέσεις εναντίον του συστήματος.

Η καλύτερη αντιμετώπιση για την εν λόγω ευπάθεια είναι η απενεργοποίηση της λειτουργίας εμφάνισης περιεχομένων. Στο παρελθόν, οι διαχειριστές έδιναν μη αναμενόμενα ονόματα στους πόρους υποθέτοντας ότι με τον τρόπο αυτό κανείς δεν θα τους εντοπίσει. Ωστόσο, η χρήση αυτού του μοντέλου, γνωστό και ως "Ασφάλεια μέσω συσκότισης" (Security Through Obscurity), δεν είναι αποτελεσματικό αφού πλέον, υπάρχουν σαρωτές ευπαθειών (vulnerability scanners) όπως είναι ο Wikto, που μπορεί να ελέγξει για ύπαρξη κρυφών πόρων με ή χωρίς την επανεξέταση του robot.txt αρχείου. [7] [12]

2.1.3 **Ακατάλληλα δικαιώματα στο σύστημα αρχείων (**Improper Filesystems Permission)

Πολλές φορές ένας εισβολέας εκμεταλλεύεται την εφαρμογή ακατάλληλων δικαιωμάτων σε αρχεία, φακέλους και συμβολικούς συνδέσμους (symbolic links ή symlinks) θέτοντας σε κίνδυνο την εμπιστευτικότητα, την ακεραιότητα και την διαθεσιμότητα μιας διαδικτυακής εφαρμογής.

Η εφαρμογή των κατάλληλων δικαιωμάτων σε ευαίσθητα αρχεία και καταλόγους ενός συστήματος μπορεί να συμβάλει στην ομαλή λειτουργία της εφαρμογής. Σε αντίθετη περίπτωση, ο εισβολέας μπορεί να διαγράψει ή να τροποποιήσει το περιεχόμενο των αρχείων ή φακέλων και να προκαλέσει σοβαρά προβλήματα στην εφαρμογή. Επιπροσθέτως, η ακατάλληλη εφαρμογή δικαιωμάτων σε συμβολικούς συνδέσμους (symbolic links ή symlinks) μπορεί να διευκολύνει έναν εισβολέα στο να κλιμακώσει τα προνόμιά του ή να αποκτήσει εξουσιοδοτημένη πρόσβαση σε ευαίσθητα αρχεία.

Η παρακάτω λίστα περιλαμβάνει μερικά από τα δικαιώματα που σχετίζονται με αρχεία:

- Read
- Write
- Modify
- Execute
- List Folder Contents
- Traverse Folder
- List Folder
- Read Attributes
- Read Extended Attributes
- Create Files/ Write Data
- Create Folders /Append Data
- Write Attributes
- Write Extended Attributes

- Delete Subfolders and Files
- Delete Read Permissions
- Change Permissions
- Take Ownership and Synchronize

Κάθε αρχείο, κατάλογος ή συμβολικός σύνδεσμος που βρίσκεται στον εξυπηρετητή, έχει ένα σύνολο από δικαιώματα που αντιστοιχούν σε κάποιο λογαριασμό του λειτουργικού συστήματος. Από την άλλη πλευρά, το σύνολο αυτών των δικαιωμάτων (για τα ίδια αρχεία, καταλόγους και συμβολικούς συνδέσμους) μπορεί να είναι διαφορετικό για κάθε λογαριασμό. Για παράδειγμα, όταν ο περιηγητής ενός ανώνυμου χρήστη ζητάει ένα αρχείο, τότε ο εξυπηρετητής αποφασίζει με ποιό τρόπο θα απαντήσει σε αυτό το αίτημα (request). Ανάλογα με τα δικαιώματα που έχουν εκχωρηθεί στο αρχείο, ο εξυπηρετητής είτε θα καταφέρει να διαβάσει το αρχείο και να επιστρέψει τα περιεχόμενά του ως απάντηση στο αίτημα ή θα επιστρέψει ένα μήνυμα σφάλματος επισημαίνοντας την άρνηση πρόσβασης (403 Forbidden). [7] [13]

2.1.4 **Ακατάλληλος χειρισμός εισόδου (**Improper Input Handling)

Ο ακατάλληλος χειρισμός εισόδου (Improper Input Handling) αφορά τη κακή διαχείριση των δεδομένων που εισάγονται σε μια διαδικτυακή εφαρμογή. Ο επιτιθέμενος μπορεί να εισάγει αυτά τα δεδομένα χειροκίνητα, με χρήση λογισμικού αλλά και με άλλους τρόπους. Η λήψη τέτοιων δεδομένων συνήθως γίνεται είτε από τη συμπλήρωση μιας φόρμας είτε μέσω URL που περιέχουν συμβολοσειρές ερωτημάτων (query strings), δεδομένων POST, επικεφαλίδες HTTP, cookies κτλ.

Η διαχείριση εισόδου (Input Handling) περιλαμβάνει τεχνικές ασφαλείας με τις οποίες μια διαδικτυακή εφαρμογή ελέγχει τα δεδομένα που εισάγονται στο διακομιστή (server) ως προς τον τύπο, το εύρος τιμών, το μήκος και το συντακτικό. Η **επικύρωση (Validation)** ή **καθαρισμός (Sanitizing)** των δεδομένων μπορεί να γίνει με τους εξής τρόπους:

1.Με την εφαρμογή **στρατηγικής τερματισμού (Terminate / stop / abort on input problems)**: Η στρατηγική αυτή είναι ασφαλής αφού εμποδίζει την εισαγωγή μη επιθυμητών χαρακτήρων με το να σταματάει την εκτέλεση των εντολών. Ωστόσο, όταν δεν μπορεί να εφαρμοστεί επαρκώς, μπορεί να κατακλύσει το σύστημα με απροσδόκητα δεδομένα, αναγκάζοντας το να δαπανήσει τους λιγοστούς πόρους επεξεργασίας και επικοινωνίας που διαθέτει με αποτέλεσμα να μην μπορεί να απαντήσει σε αιτήματα του χρήστη.

2. Με χρήση **λευκής ή μαύρης λίστας (white or black list)**: Στην ασφάλεια των υπολογιστών, υπάρχουν συχνά τα δεδομένα που χαρακτηρίζονται ως καλά για εισαγωγή - τα οποία είναι απολύτως ασφαλή για καταχώρηση. Υπάρχουν επίσης και κακοί χαρακτήρες που θέτουν σε κίνδυνο την ασφάλεια. Βάση των παραπάνω, αυτές οι δυο διαφορετικές απόψεις σχετικά με το πως θα πρέπει να αντιμετωπίζεται η εισαγωγή δεδομένων οδηγούν στα εξής:

- Η Whitelist είναι μια λίστα που γνωρίζει τι μπορούμε να εισάγουμε (input). Η Whitelist υποδεικνύει ότι π.χ. τα Α, Β και Γ είναι καλά (και όλα τα άλλα δεν είναι)
- Η Blacklist είναι μια λίστα που γνωρίζει τι δεν πρέπει να εισάγουμε. Η Blacklist είναι μια λίστα που υποδεικνύει ότι π.χ. τα Δ, Ε και Ζ δεν είναι καλά (και όλα τα άλλα είναι)

Οι επαγγελματίες ασφάλειας τείνουν να προτιμούν τις Whitelist, επειδή οι Blacklists πιθανών από λάθος να επιτρέψουν την είσοδο σε μη ασφαλές περιεχόμενο ως ασφαλές. Ωστόσο σε μερικές περιπτώσεις μια Whitelist δεν μπορεί να εφαρμοστεί εύκολα

3. Με **ενσωματωμένη υποστήριξη**: Μερικές γλώσσες προγραμματισμού έχουν ενσωματωμένη υποστήριξη για κακόβουλη εισαγωγή δεδομένων **(taint checking)**. Αυτές οι γλώσσες κατά την διάρκεια που ετοιμάζονται οι διεργασίες από τον μεταγλωττιστή (compile time) ή κατά την διάρκεια που η εφαρμογή λειτουργεί (run time) "πετούν" τις εξαιρέσεις, κάθε φορά που μια μεταβλητή προέρχεται από την εισαγωγή δεδομένων του χρήστη και θέτει σε κίνδυνο την εφαρμογή π.χ. όταν εκτελεί μια shell command.

4. Με χρήση **Κωδικοποίησης (Encode)**: Προκειμένου να αποτραπεί η εισαγωγή κακόβουλου περιεχόμενου, το οποίο μπορεί να περιέχει οτιδήποτε που μπορεί να βλάψει την βάση δεδομένων θα πρέπει **να κωδικοποιείται (encoded)**. Για παράδειγμα η SQL κωδικοποίηση: *' OR 1=1 --'* κωδικοποιείται σε *\ \'\ OR\ 1\=1\ \-\-'*

Οι τεχνικές αυτές είναι αποτελεσματικές όταν εφαρμόζονται από την πλευρά του διακομιστή (server) και όχι από την πλευρά του πελάτη (client) καθώς ο τελευταίος θεωρείται αναξιόπιστος.

Ο ακατάλληλος χειρισμός εισόδου μπορεί να προκαλέσει επιθέσεις όπως:

- Υπερχείλιση ενδιάμεσης μνήμης (Buffer Overflow)
- Άρνηση Υπηρεσίας (Denial of Service ή DoS Attack)
- Έγχυση σε εντολές SQL (SQL Injections) κ.α. επιθέσεις έγχυσης κώδικα
- Εκτέλεση εντολών λειτουργικού συστήματος (OS Commanding)
- Πλαστογράφηση αιτήσεων μεταξύ ιστοσελίδων (Cross-Site Scripting)
- Διάσχιση διαδρομής (Path Traversal ή Directory Traversal ή ../Attack)

Πιθανών να υπάρχουν και άλλες λύσεις, οι οποίες εξαρτώνται από την γλώσσα προγραμματισμού που χρησιμοποιείται και τι είδους επίθεσης προλαμβάνει. Π.χ. το htmLawed PHP script (εργαλείο) μπορεί να χρησιμοποιηθεί για να αφαιρέσει το κώδικα που προκαλεί

επιθέσεις XSS. Για την πρόληψη των SQL injection, τα παραμετροποιημένα (parameterized) queries (γνωστά και ως prepared statements και οι bind variables) είναι εξαιρετικά για την βελτίωση της ασφάλειας ενώ παράλληλα βελτιώνουν την σαφήνεια και την απόδοση του κώδικα. [7] [14] [15]

2.1.5 **Ακατάλληλος χειρισμός εξόδου (**Improper output Handling)

Με την έννοια διαχείριση εξόδου (Output Handling) αναφερόμαστε στον τρόπο με τον οποίο, μια διαδικτυακή εφαρμογή δημιουργεί δεδομένα προς έξοδο από το διακομιστή (server) σε οποιαδήποτε εξωτερική πηγή.

Η σωστή διαχείριση εξόδου (Output Handling) εμποδίζει την απροσδόκητη ή την ακούσια ερμηνεία των δεδομένων από τον χρήστη. Για την επίτευξη αυτού του στόχου, οι προγραμματιστές πρέπει να κατανοήσουν το μοντέλο δεδομένων της εφαρμογής, τον τρόπο με τον οποίο τα δεδομένα θα πρέπει να παραδίδονται σε άλλα τμήματα της εφαρμογής, και πως τελικά θα παρουσιαστούν στον χρήστη. Πάνω από όλα, θα πρέπει να αντιμετωπίζουν όλα τα δεδομένα που βρίσκονται μέσα στην εφαρμογή ως μη αξιόπιστα, και με αυτή την υπόθεση θα πρέπει να επιλέγουν τα πιθανά σενάρια εξόδου των δεδομένων. Μερικές τεχνικές για την εξασφάλιση του ορθού χειρισμού των δεδομένων εξόδου είναι η **διαδικασία διαφυγής (escaping)** χαρακτήρων και η **κωδικοποίηση (encode).**

Η **διαδικασία διαφυγής (escaping)** χαρακτήρων γίνεται με χρήση κατάλληλων τεχνικών της γλώσσας προγραμματισμού που χρησιμοποιείται. Ανάλογα με το είδος της επίθεσης που θέλει να αποφύγει ο προγραμματιστής, υπάρχει και ο κατάλληλος τρόπος. Για παράδειγμα στην PHP:

- Χρησιμοποιούμε την συνάρτηση mysql_real_escape() ή prepared statements για την προστασία από SQL Injections
- Χρησιμοποιούμε τις συναρτήσεις htmlspecialchars() ή htmlentities() για την αποφυγή Cross-site Scripting επιθέσεων.

Σε καμία περίπτωση η έννοια της **διαφυγής (escape)** χαρακτήρων δεν πρέπει να συνδέεται με την έννοια του **φιλτραρίσματος (filtering)** των δεδομένων. Με απλά λόγια, όλο το φιλτράρισμα πρέπει να γίνεται κατά την **είσοδο (input)** των δεδομένων στον διακομιστή (server) και η διαφυγή (escaping) θα πρέπει να γίνεται κατά την **έξοδο (output)** από αυτόν. [7] [14] [16]

Ο ακατάλληλος χειρισμός εισόδου μπορεί να προκαλέσει επιθέσεις όπως:

- Πλαστογράφηση περιεχομένου (Content Spoofing)
- Πλαστογράφηση αιτήσεων μεταξύ ιστοσελίδων (Cross-Site Scripting)
- Διάσπαση αιτήσεων HTTP (HTTP Response Splinting)
- Λαθραία αιτήματα HTTP (HTTP Response Smuggling)

- Έγχυση στον LDAP (LDAP Injection)
- Εκτέλεση εντολών λειτουργικού συστήματος (OS Commanding)
- Παράκαμψη δρομολόγησης (Routing Detour)
- Εκμετάλλευση πινάκων σε μηνύματα SOAP (Soap Array Abuse)
- Κατάχρηση ανακατεύθυνσης URL (URL Redirector Abuse)
- Έγχυση XML (XML Injection)
- Έγχυση XQuery (XQuery Injection)
- Έγχυση σε XPath (XPath Injection)
- Έγχυση στο σύστημα ηλεκτρονικής αλληλογραφίας (Mail Command Injection)
- Έγχυση μηδενικών bytes (Null Byte Injection)
- Έγχυση σε εντολές SQL (SQL Injections)

2.1.6 **Διαρροή πληροφοριών (**Information Leakage)

Η διαρροή πληροφοριών είναι μια ευπάθεια της εφαρμογής που αποκαλύπτει ευαίσθητα δεδομένα, όπως τεχνικές λεπτομέρειες της εφαρμογής, το περιβάλλον εγκατάστασης, τα δεδομένα των χρηστών του συστήματος κλπ. Αυτά τα δεδομένα μπορεί να υπάρχουν μέσα σε HTML σχόλια, σε μηνύματα λάθους ή απλά να είναι σε κοινή θέα και μπορούν να βοηθήσουν έναν εισβολέα στο να οργανώσει μια επίθεση.

Συχνά ένας προγραμματιστής αφήνει σχόλια στον HTML κώδικα και σε κάποια script για τον εντοπισμό σφαλμάτων (debugging) ή για να διευκολυνθεί κατά την ανάπτυξη της εφαρμογής. Αυτά τα σχόλια μπορεί να δίνουν λεπτομέρειες για το πως λειτουργεί η εφαρμογή, πως είναι γραμμένα τα SQL ερωτήματα ή σε έσχατες περιπτώσεις, να περιέχουν συνθηματικά (usernames & passwords) τα οποία χρησιμοποιήθηκαν κατά την διάρκεια δοκιμών.

Οι αριθμοί έκδοσης λογισμικού και τα μηνύματα λάθους με περιττές λεπτομέρειες είναι παραδείγματα κακής διαμόρφωσης εξυπηρετητή (Server Misconfiguration). Τα μηνύματα λάθους είναι χρήσιμα για τον εντοπισμό σφαλμάτων (debugging) και για την αντιμετώπιση δυσλειτουργιών (troubleshooting). Ωστόσο, θα πρέπει να εμφανίζονται μόνο όταν ο προγραμματιστής έχει ενεργοποιημένη την αντίστοιχη λειτουργία.

Οι διαρροή των προσωπικών δεδομένων των χρηστών μπορεί να οφείλεται σε ανεπαρκή αυθεντικοποίηση (Insufficient Authentication),σε ανεπαρκή εξουσιοδότηση (Insufficient Authorization) ή σε έλλειψη κατάλληλης κρυπτογράφησης ή έλεγχου πρόσβασης. [7] [17]

2.1.7 **Μη ασφαλής Ευρετηριασμός (**Insecure Indexing)

Όσο το περιεχόμενο των ιστοσελίδων αυξάνονταν, τόσο περισσότερο δυσκολεύονταν οι επισκέπτες να βρουν τις πληροφορίες που έψαχναν. Για να λυθεί το πρόβλημα αυτό, οι κατασκευαστές τοποθετούσαν μηχανές αναζήτησης. Μια μηχανή αναζήτησης πρώτα "μαθαίνει" την ιστοσελίδα, εξετάζοντας τις σελίδες της και συνδέει λέξεις-κλειδιά (keywords) ενημερώνοντας την εσωτερική βάση δεδομένων. Αυτή η διαδικασία ονομάζεται ευρετηριασμός (indexing). Η διαδικασία δημιουργίας ευρετηρίου εκτελείται συνεχώς προκειμένου να εμφανίζει τα σωστά αποτελέσματα στον χρήστη.

Υπάρχουν δύο είδη ευρετηρίασης:

- Η εξ αποστάσεως (web / HTTP based), που χρησιμοποιεί απομακρυσμένες (3rd party) μηχανές αναζήτησης όπως της Google και της Yahoo, οι οποίες διασχίζουν την εφαρμογή από σελίδα σε σελίδα, ξεκινώντας συνήθως από την αρχική σελίδα και συνεχίζοντας αναδρομικά ακλουθώντας συνδέσμους.
- Η τοπική, που έχει άμεση πρόσβαση στον εξυπηρετητή και στο σύστημα αρχείων (file system), ευρετηριάζοντας την ιστοσελίδα με τη μετάβαση σε όλα τα αρχεία, μέχρι και στη ρίζα του συστήματος.

Ο μη ασφαλής ευρετηριασμός είναι μια απειλή που θέτει σε κίνδυνο την εμπιστευτικότητα των δεδομένων μιας ιστοσελίδας. Αυτό συμβαίνει συνήθως όταν χρησιμοποιείται η τοπική μέθοδος ευρετηρίασης κατά την οποία συμπεριλαμβάνονται αρχεία που περιέχουν ευαίσθητες πληροφορίες. Η διαρροή των πληροφοριών μπορεί να γίνει με την εκτέλεση μιας σειράς ερωτημάτων στη μηχανή αναζήτησης. Μπορεί να μην είναι τόσο εύκολο για τον επιτιθέμενο να παρακάμψει το μοντέλο ασφαλείας της μηχανής αναζήτησης αλλά αν συμβεί κάτι τέτοιο, είναι πολύ δύσκολο να εντοπιστεί και να αποτραπεί μια επίθεση αυτού του είδους. [7] [18]

2.1.8 **Ανεπαρκή μέτρα έναντι αυτοματοποιημένης εκτέλεσης (**Insufficient Anti-automation)

Η ευπάθεια αυτή συμβαίνει όταν μια εφαρμογή διαδικτύου επιτρέπει σε έναν εισβολέα να αυτοματοποιήσει μια διαδικασία, η οποία θα έπρεπε να εκτελεστεί χειροκίνητα. Μια τέτοια διαδικασία μπορεί να είναι η συμπλήρωση μιας φόρμας εισόδου, επικοινωνίας, εγγραφής, σχολίων κλπ

Αν δεν υπάρχει ο κατάλληλος έλεγχος, αυτοματοποιημένα robot ή εισβολείς μπορούν να χρησιμοποιήσουν επανειλημμένα κάποια από τις λειτουργίες της εφαρμογής για να επηρεάσουν την ομαλή λειτουργία του συστήματος. Ένα αυτοματοποιημένο robot θα μπορούσε δυνητικά να εκτελέσει χιλιάδες αιτήματα σε ένα λεπτό, προκαλώντας επιθέσεις ωμής βίας (brute force) ή επίθεση άρνησης υπηρεσίας (Denial of Service).

Για την προστασία από αυτή την ευπάθεια χρησιμοποιούνται κάποια δωρεάν πρόσθετα (plug-ins), το Google reCAPTCHA είναι ένα από αυτά. [7] [19]

2.1.9 **Ανεπαρκής αυθεντικοποίηση** (Insufficient Authentication)

Η ευπάθεια της ανεπαρκούς αυθεντικοποίησης συμβαίνει, όταν μια ιστοσελίδα επιτρέπει σε έναν εισβολέα να αποκτήσει πρόσβαση σε δεδομένα και λειτουργίες, χωρίς προηγουμένως να έχει προβεί σε σωστό και πλήρη έλεγχο της ταυτότητάς του.

Το γεγονός αυτό συνήθως οφείλονταν στην εφαρμογή μιας τεχνικής που ήταν γνωστή ως "Ασφάλεια μέσω συσκότισης" (Security Through Obscurity). Σύμφωνά με αυτή την τεχνική, οι πόροι μιας εφαρμογής δεν συνδέονταν με την κύρια ιστοσελίδα με κάποιο σύνδεσμο και προστατεύονταν με την απόκρυψη της θέσης τους. Ο μόνος τρόπος προσπέλασης τους ήταν να γνωρίζει κάποιος την ακριβή διεύθυνση URL. Το να ανακαλύψει ένας εισβολέας αυτή την διεύθυνση, ήταν σχετικά εύκολο και επιτυγχάνονταν με μια επίθεση ωμής βίας (brute force attack) σε ένα αρχείο το οποίο διαθέτει κοινότυπο όνομα και κοινότυπο φάκελο τοποθεσίας, όπως για παράδειγμα ο φάκελος "/admin" καθώς και οι φάκελοι αποθήκευσης μηνυμάτων λάθους, αρχείων καταγραφής, αρχείων βοήθειας κ.λπ. [7] [20]

2.1.10 **Ανεπαρκής εξουσιοδότηση** (Insufficient Authorization)

Σύμφωνα με την πολιτική ασφαλείας μιας εφαρμογής, σε κάθε τύπο χρήστη αναλογούν και κάποια δικαιώματα (permissions). Για παράδειγμα, ένας απλός επισκέπτης μπορεί να δει το περιεχόμενο μιας σελίδας, ένας εγγεγραμμένος χρήστης μπορεί να λαμβάνει ενημερωτικά email σχετικά με νέες πληροφορίες και ο διαχειριστής μπορεί να έχει πρόσβαση στην διαχείριση του περιεχομένου. Σε καμία περίπτωση δεν πρέπει να έχουν πρόσβαση στην διαχείριση του περιεχομένου οι δυο πρώτοι τύποι χρηστών.

Η ευπάθεια της ανεπαρκούς εξουσιοδότησης συμβαίνει, όταν μια εφαρμογή δεν διενεργεί τους κατάλληλους ελέγχους εξουσιοδότησης για να εξασφαλίσει ότι ένας χρήστης εκτελεί μια λειτουργία ή έχει πρόσβαση στα δεδομένα της εφαρμογής, κατά τρόπο σύμφωνο με την πολιτική ασφαλείας αυτής. [7] [21]

Σύμφωνα με τον OWASP η εν λόγω ευπάθεια καλύπτεται με τις παρακάτω έννοιες:

- ✓ Ανεπαρκής περιορισμός πρόσβασης URL (Failure to Restrict URL Access)
- ✓ Επισφαλής άμεση αναφορά αντικειμένου (Insecure Direct Object References)

2.1.11 **Ανεπαρκής διαδικασία ανάκτησης συνθηματικών (**Insufficient Password Recovery)

Όταν ένας χρήστης επιθυμεί να εγγραφεί (sign up) σε μια ιστοσελίδα καλείται να συμπληρώσει μια φόρμα με τα στοιχεία του. Μεταξύ αυτών των στοιχείων είναι το email, ο μυστικός κωδικός πρόσβασης (password), το όνομά του (username) και άλλες προσωπικές πληροφορίες. Κάθε φορά που θέλει να συνδεθεί (log in) σε αυτή την ιστοσελίδα το μόνο που πρέπει να κάνει είναι να πληκτρολογήσει σωστά το μυστικό κωδικό πρόσβασης (password) και το όνομά του (username) προκειμένου να πιστοποιήσει την ταυτότητά του και να αποκτήσει πρόσβαση σε πληροφορίες και λειτουργίες που διαφορετικά δεν θα μπορούσε.

Πολλές φορές, για διάφορους λόγους, οι χρήστες δεν συγκρατούν ή ξεχνούν τον προσωπικό κωδικό πρόσβασής τους με αποτέλεσμα να μην μπορούν να συνδεθούν και να πιστοποιήσουν την ταυτότητα τους. Το γεγονός αυτό, οδήγησε τους προγραμματιστές διαδικτυακών εφαρμογών στην λειτουργία αυτόματης ανάκτησης του κωδικού πρόσβασης (password) των χρηστών μέσω ειδικής εφαρμογής στις ιστοσελίδες αυτές.

Η ευπάθεια ανεπαρκούς διαδικασίας ανάκτησης συνθηματικών (Insufficient Password Recovery) παρατηρείται σε μια διαδικτυακή εφαρμογή όταν επιτρέπει σε έναν εισβολέα να αλλάξει ή να αποκτήσει τον κωδικό πρόσβασης (password) ενός τρίτου χρήστη εξουδετερώνοντας το μηχανισμό ανάκτησης συνθηματικών.

Για την σωστή ανάκτηση συνθηματικών υπάρχουν αρκετές αυτοματοποιημένες διαδικασίες. Η πιο γνωστή, είναι η απάντηση σε μια "μυστική ερώτηση" η οποία τέθηκε στον χρήστη κατά τη διαδικασία εγγραφής του (sign up). Η ερώτηση αυτή, επιλέχθηκε από τον ίδιο τον χρήστη μέσα από μια λίστα έτοιμων ερωτήσεων ή πληκτρολογήθηκε από τον ίδιο. Ένας άλλος μηχανισμός που χρησιμοποιείται ευρέως, είναι η τεχνική κατά την οποία παρέχεται στο χρήστη ένα "hint" κατά την εγγραφή του, που πιθανόν να τον βοηθήσει μελλοντικά σε περίπτωση απώλειας του κωδικού του (password). Άλλοι μηχανισμοί ανάκτησης απαιτούν από το χρήστη να επαληθεύσει κάποια από τα στοιχεία των προσωπικών του δεδομένων, όπως για παράδειγμα τον αριθμό τηλεφώνου, τη διεύθυνσή του κα., τα οποία είχε εισάγει κατά την εγγραφή του (sign up). Στην συνέχεια, εφόσον ο χρήστης αποδείξει ποιος είναι, το σύστημα ανάκτησης θα αποστείλει στο email του ένα νέο κωδικό πρόσβασης.

Ένα σύστημα ανάκτησης κωδικού πρόσβασης μπορεί να τεθεί σε κίνδυνο με επιθέσεις ωμής βίας (brute force attacks). [7] [22]

2.1.12 **Ανεπαρκής επικύρωση διαδικασιών** (Insufficient Process Validation)

Αυτή η ευπάθεια συμβαίνει σε εφαρμογές όπου ο εισβολέας μπορεί να παρακάμψει την προβλεπόμενη λογική ροή ή την λογική της εφαρμογής. Ο έλεγχος της ροής (flow control) και ο τύπος της επιχειρηματικής λογικής (business logic) είναι δύο κύριοι τύποι διαδικασιών που απαιτούν επικύρωση.

Ο **έλεγχος ροής** περιλαμβάνει διαδικασίες πολλαπλών βημάτων που πρέπει να εκτελούνται με συγκεκριμένη σειρά από το χρήστη. Τέτοιες διαδικασίες είναι η δημιουργία λογαριασμού, η ολοκλήρωση μιας ηλεκτρονικής αγοράς, η ανάκτηση του κωδικού πρόσβασης κ.α.. Σε κάθε βήμα, ο χρήστης πρέπει να εκτελεί μια συγκεκριμένη ενέργεια. Όταν ένας εισβολέας εκτελέσει ένα βήμα λανθασμένα ή εκτός σειράς, ο έλεγχος πρόσβασης μπορεί να παρακαμφθεί και να εμφανιστεί ένα σφάλμα στην εφαρμογή.

Η **επιχειρηματική λογική**, αναφέρεται στο πλαίσιο εκείνο το οποίο μια διαδικασία θα εκτελεστεί, όπως προβλέπεται από τις επιχειρηματικές απαιτήσεις. Η αξιοποίηση των αδυναμιών της επιχειρηματικής λογικής απαιτεί γνώση της επιχείρησης. Μια εφαρμογή συνήθως πάσχει από επιχειρηματική λογική όταν ο επιτιθέμενος την εκμεταλλεύεται χωρίς να γνωρίζει πολλά πράγματα για αυτή.

Για την αντιμετώπιση αυτής της ευπάθειας πρέπει να λαμβάνονται τυπικά μέτρα ασφαλείας, όπως έλεγχοι και επιθεωρήσεις του κώδικα. Μια προσέγγιση δοκιμών αυτού του είδους προσφέρεται και στον οδηγό testing του OWASP. [7] [23]

2.1.13 **Ανεπαρκής αυτόματη λήξη συνόδων** (Insufficient Session Expiration)

Ένα πρωτόκολλο άνευ μνήμης (stateless), όπως το HTTP, δεν απαιτεί από το διακομιστή (server) να διατηρεί τις πληροφορίες συνόδου (session) ή τη κατάσταση επικοινωνίας για κάθε έναν (συνδεδεμένο) χρήστη καθ' όλη την διάρκεια των πολλαπλών αιτήσεων (requests). Για αυτό το λόγο, οι διαδικτυακές εφαρμογές χρησιμοποιούν τα cookies, που αποθηκεύουν αυτά τα αναγνωριστικά συνόδου, και προσδιορίζουν μονοσήμαντα τους χρήστες και τα αιτήματα τους (requests). Τα cookies δεν αποθηκεύονται από την πλευρά του εξυπηρετητή (server) αλλά από την πλευρά του χρήστη (client).

Η ευπάθεια της ανεπαρκούς λήξης συνόδου παρατηρείται όταν ένας εισβολέας εκμεταλλεύεται την διάρκεια ζωής των διαπιστευτηρίων (credentials) ή αναγνωριστικών συνόδου, ενός συνδεδεμένου χρήστη, για την αυθεντικοποίηση του. Αυτή η ευπάθεια μπορεί να οδηγήσει σε επιθέσεις που κλέβουν ή επαναχρησιμοποιούν τα αναγνωριστικά εισόδου των συνδεδεμένων χρηστών-θυμάτων (**session hijacking**).

Υπάρχουν δύο τύποι λήξης συνόδου:

- **Το χρονικό όριο αδράνειας (inactivity)**. Με τον όρο αυτό ορίζουμε ένα χρονικό όριο, ως μέγιστο επιτρεπτό χρόνο αδράνειας για κάθε σύνοδο (session) πριν αυτή ακυρωθεί.
- **Το απόλυτο χρονικό όριο (absolute)**. Ως απόλυτο χρονικό όριο ορίζουμε το συνολικό χρόνο για τον οποίο επιτρέπεται μία σύνοδος να είναι έγκυρη, χωρίς να χρειαστεί η εκ νέου επαλήθευση της ταυτότητας του χρήστη.

Στις διαδικτυακές εφαρμογές, μια συνεδρία (session) θα πρέπει να τερματίζεται είτε από τον ίδιο τον χρήστη με χρήση λειτουργίας αποσύνδεσης (Log out), είτε αυτόματα μετά από ένα προκαθορισμένο χρονικό όριο αδράνειας. [7] [24]

2.1.14 **Ανεπαρκής προστασία επιπέδου μεταφοράς** (Insufficient Transport Layer Protection)

Η ανεπαρκής προστασία επιπέδου μεταφοράς είναι μια αδυναμία ασφάλειας που προκαλείται σε εφαρμογές που δεν εφαρμόζουν κανένα μέτρο για την προστασία των πληροφοριών σε επίπεδο δικτύου, κατά την επικοινωνία μεταξύ του χρήστη και της εφαρμογής. Για την πιστοποίηση, οι εφαρμογές πιθανόν να χρησιμοποιούν πρωτόκολλο Secure Socket Layer (SSL) ή Transport Layer Security (TLS), αλλά συχνά αποτυγχάνουν να κάνουν σωστή χρήση αυτών, αφήνοντας έτσι τα δεδομένα και τα αναγνωριστικά συνεδριών (session id) εκτεθειμένα.

Σύμφωνα με τον OWASP: "Οι εφαρμογές συχνά αποτυγχάνουν να κάνουν έλεγχο ταυτότητας, να κρυπτογραφήσουν, και να προστατεύσουν την αυθεντικότητα και την εμπιστευτικότητα σε επίπεδο δικτύου. Όταν το κάνουν, μερικές φορές χρησιμοποιούν αδύναμους αλγόριθμους κρυπτογράφησης, ληγμένα ή μη έγκυρα πιστοποιητικά, ή απλά τα χρησιμοποιούν λάθος." Η λανθασμένη χρήση SSL/ TLS, μπορεί να κάνει την εφαρμογή ευάλωτη σε υποκλοπές κυκλοφορίας και τροποποίησης.

Επειδή πολλές εκδόσεις των πρωτόκολλων SSL / TLS χρησιμοποιούνται ευρέως σε πολλές εφαρμογές όπως σε ηλεκτρονικό ταχυδρομείο, σε ανταλλαγές άμεσων μηνυμάτων και σε πολλές άλλες εφαρμογές που επικοινωνούν μέσω του διαδικτύου η ευπάθεια αυτή είναι στην ένατη θέση του Top 10 κινδύνων του OWASP.

Ευτυχώς, η ανεπαρκής προστασία επιπέδου μεταφοράς είναι εύκολο να αποφευχθεί. Με χρήση ισχυρών αλγόριθμων κρυπτογράφησης μπορούν να αποφευχθούν επιθέσεις υποκλοπής, έγχυσης (injection) και ανακατεύθυνσης (redirection). Επιπροσθέτως, όλο το περιεχόμενο μιας ασφαλούς ιστοσελίδας, είτε είναι αρχεία κώδικα όπως html, JavaScript κλπ. είτε άλλα αρχεία, όπως εικόνες, video κ.α., θα πρέπει να εξυπηρετούνται μέσω HTTPS. Στην περίπτωση που χρησιμοποιείται προστασία μεταφοράς (https) και ταυτόχρονα υπάρχουν αρχεία που εξυπηρετούνται σε HTTP, τότε μιλάμε για χρήση μεικτού περιεχομένου, πράγμα που σημαίνει ότι η εφαρμογή είναι ευάλωτη σε επιθέσεις. Σε αυτή την περίπτωση ο εισβολέας μπορεί με απλή αντικατάσταση αρχείων να προκαλέσει πρόβλημα στην εφαρμογή. [7] [25] [26]

2.1.15 **Επισφαλής διαμόρφωση εξυπηρετητή (**Server Misconfiguration)

Είναι πολύ σημαντικό για έναν προγραμματιστή διαδικτυακών εφαρμογών να ελέγχει τις ρυθμίσεις του εξυπηρετητή (server) που χρησιμοποιεί. Πιο συγκεκριμένα οφείλει: [7] [27]

- Να κρατάει τον εξυπηρετητή πάντα ενημερωμένο.

- Να αποθηκεύει σε διαφορετικό σκληρό δίσκο από αυτόν του συστήματος τους διαδικτυακούς φακέλους και τα αρχεία (όπου είναι εφικτό). Με αυτό τον τρόπο θα είναι λιγότερο ευάλωτο το σύστημα σε μια επίθεση διάσχισης διαδρομής (Path Traversal ή Directory Traversal ή ../Attack)
- Να ελέγχει συχνά για επικίνδυνες και ύποπτες αιτήσεις (requests) που μπορεί να έχουν καταγραφεί (στα log files) και να μπλοκάρει τις ip διευθύνσεις που τις προκαλούν
- Να απενεργοποιεί περιττές λειτουργίες ή λειτουργίες στις οποίες έχει πρόσβαση μόνο ο διαχειριστής, όπως λειτουργίες εντοπισμού σφαλμάτων (debugging)
- Να μην αποκαλύπτει πληροφορίες για την δομή της εφαρμογής και τον τύπο του συστήματος σε αιτήσεις και σε αποκρίσεις μετά από σφάλματα. Πολλές φορές τα μηνύματα λάθους μπορεί να χρησιμοποιηθούν από τους hackers για να διαμορφώσουν το πλαίσιο της επόμενης επίθεσης.
- Να ελέγχουν τις προεπιλεγμένες ρυθμίσεις, τα δικαιώματα (permissions) των λογαριασμών σε αρχεία και καταλόγους, τα προεπιλεγμένα πιστοποιητικά (την χρήση SSL πιστοποιητικού), τις ρυθμίσεις κρυπτογράφησης και την επικοινωνία με τυχόν εξωτερικά συστήματα.
- Να κάνει εκτίμηση για την κίνηση (traffic) που αναμένεται να υπάρξει προκειμένου να μην παρατηρηθεί αδυναμία εξυπηρέτησης (Denial of Service).

2.2 **Επιθέσεις** (Attacks)

2.2.1 **Κατάχρηση της λειτουργικότητας (**Abuse of Functionality)

Η κατάχρηση της λειτουργικότητας είναι μια επίθεση που χρησιμοποιεί τα χαρακτηριστικά και τις λειτουργίες της ίδιας της εφαρμογής, με σκοπό την κατανάλωση όλων των διαθέσιμων πόρων, την εξαπάτηση των χρηστών, ή την παράκαμψη των μηχανισμών ελέγχου πρόσβασης. Μερικές λειτουργίες, συμπεριλαμβανομένων των μηχανισμών ασφαλείας, μπορεί να καταχραστούν και να προκαλέσουν απροσδόκητη συμπεριφορά, να ενοχλήσουν τους χρήστες, ή να εξαπατήσουν το σύστημα εντελώς.

Πολλές φορές, ο εισβολέας εκμεταλλεύεται λειτουργίες που έχουν να κάνουν με την αποστολή πολλαπλών email σε χρήστες. Αν ο εισβολέας καταφέρει να ελέγξει τα πεδία του μηνύματος (από, προς, θέμα και κυρίως σώμα) και δεν υπάρχουν αυτόματοι μηχανισμοί ελέγχου, τότε, οι λειτουργίες email μπορούν να μετατραπούν σε μηχανισμούς μεταφοράς ανεπιθύμητης αλληλογραφίας.

Επιπροσθέτως, θα πρέπει να δίνεται ιδιαίτερη προσοχή στη χρήση υπηρεσιών, όπως στη χρήση της υπηρεσίας μετάφρασης της Google, επειδή μπορεί να χρησιμοποιηθεί καταχρηστικά και να λειτουργήσει ως ένας ανοιχτός εξυπηρετητής αντιπροσώπευσης (proxy server).

Μια επίθεση κατάχρησης λειτουργικότητας μπορεί να συνδυαστεί με επιθέσεις όπως:

- Cross-Site Scripting (XSS)
- Έγχυση σε εντολές SQL (SQL Injection)
- Άρνηση υπηρεσίας (Denial of Service)

Αυτή η επίθεση μπορεί να συνδυαστεί και με άλλες επιθέσεις αλλά και με την ευπάθεια της ανεπαρκούς διαδικασίας ανάκτησης συνθηματικών.

Για την αντιμετώπιση αυτής της ευπάθειας, σύμφωνα με τον WASC, θα πρέπει να δίνεται ιδιαίτερη προσοχή κατά το στάδιο του σχεδιασμού της εφαρμογής από τους προγραμματιστές. [28] [29]

2.2.2 **Επιθέσεις ωμής βίας** (Brute Force)

Μια επίθεση ωμής βίας είναι μια αυτοματοποιημένη διαδικασία κατά την οποία γίνονται απεριόριστες δοκιμές προκειμένου ο επιτιθέμενος να μαντέψει το όνομα χρήστη (username), τον κωδικό πρόσβασης (password), τον αριθμό της πιστωτικής κάρτας ή το κρυπτογραφικό κλειδί. Πολλά συστήματα είναι ευάλωτα σε αυτή την επίθεση λόγω του ότι χρησιμοποιούν αδύναμα συνθηματικά ή κρυπτογραφικό κλειδί μικρού μήκους.

Όταν για παράδειγμα, ο επιτιθέμενος προσπαθεί να ανακαλύψει τον κωδικό πρόσβασης του χρήστη-θύματος προκειμένου να εισέλθει σε μια εφαρμογή, χρησιμοποιεί μια μεγάλη λίστα από λέξεις ή φράσεις σαν πιθανούς κωδικούς. Οι κωδικοί μπορεί να περιλαμβάνουν μεταβολές στις λέξεις όπως για παράδειγμα την αντικατάσταση του "a" με το "@", του "o" με το "0" κλπ. Αυτή η λίστα είναι ένα είδος λεξικού και είναι συνηθισμένη η χρήση του σε τέτοιου είδος επιθέσεις. Για το λόγο αυτό η επίθεση αυτή είναι γνωστή και ως επίθεση λεξικού (dictionary attack).

Πρακτικά η επίθεση σταματά μόλις βρεθεί το ζητούμενο δεδομένο χωρίς να χρειαστεί περαιτέρω ενημέρωση η χρησιμοποιούμενη λίστα /λεξικό. Υπάρχουν δύο τύποι επίθεσης ωμής βίας, ή κανονική (normal) και η αντίστροφη (reverse). Μια κανονική επίθεση ωμής βίας συνδυάζει ένα συγκεκριμένο όνομα χρήστη (username) με πάρα πολλούς κωδικούς πρόσβασης (passwords). Μια αντίστροφη επίθεση ωμής βίας συνδυάζει πολλά ονόματα χρηστών με έναν κωδικό πρόσβασης. Σε συστήματα με εκατομμύρια λογαριασμούς χρηστών, οι πιθανότητες να έχουν περισσότεροι του ενός χρήστη τον ίδιο κωδικό πρόσβασης είναι δραματικά πολλές.

Μια επίθεση ωμής βίας μπορεί να εκμεταλλευτεί την ευπάθεια της ανεπαρκούς διαδικασίας ανάκτησης συνθηματικών. Για παράδειγμα, αν ο επιτιθέμενος προσποιηθεί ότι είναι ο χρήστης-θύμα και ζητήσει την επανάκτηση των συνθηματικών του θα πρέπει να απαντήσει σε μια ερώτηση ασφαλείας. Σε περίπτωση που η ερώτηση είναι του τύπου: "ποιο είναι το αγαπημένο σας χρώμα" τότε ο εισβολέας μπορεί να μαντέψει εύκολα την απάντηση χωρίς να κάνει χρήση λεξικού (από τη στιγμή που ο αριθμός των χρωμάτων είναι περιορισμένος και οι πιθανότητες να απαντήσει σωστά είναι αρκετές).

Για την αντιμετώπιση αυτής της επίθεσης θα πρέπει η εφαρμογή να είναι σχεδιασμένη με τέτοιον τρόπο που: [30] [31] [32] [33]

- ✓ Να αναγκάζει τους χρήστες να χρησιμοποιούν σύνθετους κωδικούς πρόσβασης.
- ✓ Να υπάρχει περιορισμός του αριθμού των φορών που ένας χρήστης μπορεί να εισάγει λανθασμένα τον κωδικό πρόσβασής του και όταν ο χρήστης υπερβαίνει το όριο αυτό, θα πρέπει ο λογαριασμός του να μπλοκάρεται αυτόματα τουλάχιστον για ένα χρονικό διάστημα.
- ✓ Να επιτρέπει σε χρήστες με πολλά προνόμια να συνδέονται μόνο από επιλεγμένες IP διευθύνσεις
- ✓ Να χρησιμοποιούν μηχανισμούς CAPTCHA που προλαμβάνουν τις αυτοματοποιημένες επιθέσεις.

Φυσικά, υπάρχουν και άλλα παρόμοια αντίμετρα που μπορεί να συμπεριλάβει ένας προγραμματιστής στην εφαρμογή για να την προστατεύσει. Αυτοί οι μηχανισμοί καταγράφουν:

- ✓ Τις IP διευθύνσεις από τις οποίες συνδέονται πολλοί χρήστες
- ✓ Τις IP διευθύνσεις από τις οποίες γίνονται πολλές αποτυχημένες προσπάθειες σύνδεσης (login)
- ✓ Τα ονόματα των χρηστών που συνδέονται από διαφορετικές IP διευθύνσεις

2.2.3 **Υπερχείλιση Ενδιάμεσης Μνήμης** (Buffer Overflow / Buffer overrun)

Η υπερχείλιση ενδιάμεσης μνήμης είναι μια μορφή επίθεσης που συμβαίνει όταν σε μια περιοχή μνήμης καταγράφονται περισσότερα δεδομένα από όσα μπορεί να διαχειριστεί. Ένας τρόπος για να πραγματοποιηθεί αυτή η επίθεση είναι η εισαγωγή δεδομένων που περιλαμβάνουν εντολές κώδικα. Η επίθεση αυτή επιτρέπει στον εισβολέα να τροποποιεί τα σημεία που θέλει από τις διευθύνσεις της μνήμης που στοχεύει.

Η υπερχείλιση ενδιάμεσης μνήμης μπορεί να οδηγήσει σε άρνηση υπηρεσίας (Denial of Service) όταν η μνήμη έχει καταστραφεί. Σε άλλες περιπτώσεις, η εν λόγω επίθεση μπορεί να επηρεάσει τον τρόπο λειτουργίας της εφαρμογής προκαλώντας τροποποίηση των εσωτερικών μεταβλητών, κατάρρευση ή έλεγχο της εκτέλεσης της επεξεργασίας.

Από την στιγμή που ο εισβολέας πρέπει να εκμεταλλευτεί προσαρμοσμένο κώδικα σε ένα απομακρυσμένο σύστημα, θα πρέπει να εκτελέσει μια "τυφλή" επίθεση, κάτι που καθιστά πολύ δύσκολη την επιτυχία της.

Για την αποφυγή τέτοιων επιθέσεων θα πρέπει να χρησιμοποιούνται γλώσσες προγραμματισμού υψηλού επιπέδου που να μην επιτρέπουν την εκτέλεση εντολών απευθείας στη μνήμη. Επιπλέον, θα πρέπει να πραγματοποιούνται έλεγχοι στα δεδομένα που εισάγονται από το χρήστη. [34]

2.2.4 **Πλαστογράφηση περιεχομένου (**Content Spoofing)

Η πλαστογράφηση περιεχομένου είναι μια τεχνική επίθεσης, η οποία χρησιμοποιείται από έναν εισβολέα προκειμένου να παρουσιάσει μια πλαστή ή τροποποιημένη ιστοσελίδα σε έναν χρήστη ως νόμιμη. Σύμφωνα με τον OWASP, η επίθεση αυτή αναφέρεται ως έγχυση περιεχομένου (content injection) και άλλες φορές ως εικονική παραμόρφωση (virtual defacement).

Η επίθεση αυτή πολλές φορές χρησιμοποιείται σε συνδυασμό με τη κοινωνική δικτύωση (π.χ. μέσω email ή ιστοσελίδων κοινωνικής δικτύωσης). Έχει σαν βάση την εκμετάλλευση του κώδικα και την εμπιστοσύνη του θύματος ως προς την ιστοσελίδα. Ο εισβολέας μερικές φορές χρησιμοποιεί απλό κείμενο ενώ σε άλλες περιπτώσεις μπορεί να τροποποιήσει τις πληροφορίες και τους συνδέσμους, καθώς και άλλες html ετικέτες, μέσα από τον εξυπηρετητή της εφαρμογής. Σε μια φαινομενικά αξιόπιστη ιστοσελίδα, το θύμα δεν αντιλαμβάνεται την απάτη αφού οπτικά δεν υπάρχει καμία διαφορά και το URL που εμφανίζεται στον περιηγητή φαίνεται να είναι νόμιμο.

Ουσιαστικά, η επίθεση αυτή βασίζεται στην ευπάθεια του ακατάλληλου χειρισμού εισόδου και εξόδου των δεδομένων που εισάγει ο χρήστης μέσα σε φόρμες συμπλήρωσης στοιχείων ή φόρμες σύνδεσης. Τις περισσότερες φορές, ο στόχος της επίθεσης είναι η υποκλοπή των προσωπικών δεδομένων του θύματος, όπως για παράδειγμα ο αριθμός της πιστωτικής του κάρτας, τα συνθηματικά του (username & password) κα.

Η πλαστογράφηση περιεχομένου είναι μια επίθεση παρόμοια με την Cross-Site Scripting (XSS) με τη διαφορά ότι η δεύτερη χρησιμοποιεί JavaScript κώδικα. Το γεγονός ότι σε μια εφαρμογή χρησιμοποιούνται αντίμετρα για τη προστασία από XSS επιθέσεις δεν σημαίνει ότι προστατεύεται και από επιθέσεις πλαστογράφησης περιεχομένου που βασίζονται σε απλό κείμενο.

Για την αποφυγή αυτής της επίθεσης θα πρέπει να δίνεται ιδιαίτερη προσοχή κατά το στάδιο της υλοποίησης μιας εφαρμογής. Αυτό απαιτεί: [35]

- ✓ Την επικύρωση των δεδομένων που εισάγει ο χρήστης.
- ✓ Τη σωστή κωδικοποίηση των δεδομένων του χρήστη που εξάγονται από την εφαρμογή
- ✓ Τη χρήση της μεθόδου POST, αν είναι δυνατόν
- ✓ Τη χρήση διαδικτυακών σαρωτών, π.χ. Acunetix, για τυχόν ευπάθειες στον εξυπηρετητή

2.2.5 **Πρόβλεψη διαπιστευτηρίων/συνόδου** (Credential / Session Prediction)

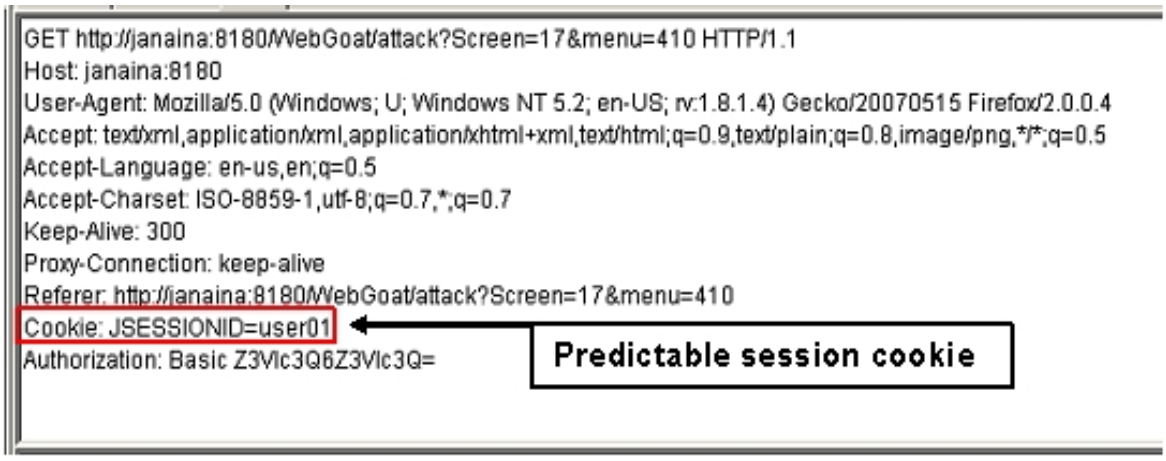
Όταν μια ιστοσελίδα απαιτεί από έναν χρήστη να πιστοποιήσει την ταυτότητα του, συνήθως ζητά από αυτόν να εισάγει τα συνθηματικά / διαπιστευτήριά του (username & password). Αφού γίνει η επαλήθευση των συνθηματικών, η ιστοσελίδα δημιουργεί ένα μοναδικό αναγνωριστικό συνεδρίας (session ID) προκειμένου να πιστοποιήσει ότι η συνεδρία του χρήστη είναι αυθεντική.

Η επίθεση πρόβλεψης διαπιστευτηρίων /συνόδου εστιάζει στη πρόβλεψη των τιμών που χρησιμοποιούνται σε αναγνωριστικά συνεδρίας (session ID) και επιτρέπει σε έναν εισβολέα να παρακάμψει τους μηχανισμούς ελέγχου ταυτότητας των χρηστών. Με την ανάλυση και κατανόηση της διαδικασίας αναπαραγωγής των session ID, ο εισβολέας μπορεί να προβλέψει μια έγκυρη τιμή για ένα session ID και να αποκτήσει πρόσβαση σε μια εφαρμογή.

Η εν λόγω επίθεση είναι γνωστή ως πειρατεία συνεδρίας (**session hijacking**) αλλά και ως cookie hijacking. Η ονομασία cookie hijacking χρησιμοποιείται επειδή μερικές φορές ο εισβολέας προσπαθεί να κλέψει τα cookies, τα οποία χρησιμοποιούνται για να διατηρήσουν μια συνεδρία (session) που έχει εγκατασταθεί μεταξύ ενός χρήστη και του εξυπηρετητή. Τα cookies μπορούν εύκολα να κλαπούν από τον εισβολέα είτε χρησιμοποιώντας έναν ενδιάμεσο υπολογιστή είτε με την πρόσβαση στα cookies που βρίσκονται στον υπολογιστή του θύματος.

Το πρώτο πράγμα που χρειάζεται ένας εισβολέας είναι να συλλέξει μερικές έγκυρες session ID τιμές που χρησιμοποιούνται για την πιστοποίηση των χρηστών. Στη συνέχεια, πρέπει να κατανοήσει την δομή μιας session ID τιμής, τις πληροφορίες που χρησιμοποιούνται για την κατασκευή της, καθώς και τον αλγόριθμο κρυπτογράφησης και κατακερματισμού για την προστασία της. Μερικές μη αξιόπιστες εφαρμογές χρησιμοποιούν για τη σύνθεση των session ID τιμών το όνομα χρήστη (username) καθώς και άλλες προβλέψιμες πληροφορίες, όπως

χρονοσφραγίδες (timestamp) ή την IP διεύθυνση του χρήστη. Στη χειρότερη περίπτωση, η πληροφορίες αυτές χρησιμοποιούνται σε μορφή απλού κειμένου ή κωδικοποιούνται με χρήση κάποιου αδύναμου αλγόριθμου (π.χ. κωδικοποίηση σε base64).



Εικόνα 2.1 - Predictable session cooking [151]

Υπάρχουν τέσσερις κύριες μέθοδοι που χρησιμοποιούνται για τη διάπραξη αυτής της επίθεσης. Αυτές είναι:

- ✓ **Ο κακόβουλος ορισμός αναγνωριστικών συνόδου (Session fixation)**, όπου ο επιτιθέμενος ορίζει το session ID ενός χρήστη έτσι ώστε να είναι γνωστό σε αυτόν.
- ✓ **Session sidejacking**, όπου ο επιτιθέμενος χρησιμοποιεί λογισμικό για την παρακολούθηση της κυκλοφορίας του δικτύου (packet sniffing) μεταξύ του χρήστη και της εφαρμογής για να κλέψει το session cookie. Πολλές ιστοσελίδες χρησιμοποιούν SSL κρυπτογράφηση μόνο στην σελίδα που γίνεται η σύνδεση (login) προκειμένου να εμποδίσουν τους επιτιθέμενους να δουν τον κωδικό πρόσβασης, αλλά δεν χρησιμοποιούν κρυπτογράφηση σε όλο το υπόλοιπο μέρος της ιστοσελίδας μετά την πιστοποίηση. Αυτό επιτρέπει στους επιτιθέμενους, που μπορούν να διαβάσουν την κίνηση του δικτύου, να υποκλέψουν όλα τα δεδομένα που έχουν υποβληθεί στον εξυπηρετητή ή φαίνονται στις σελίδες από τον χρήστη (client). Από την στιγμή που αυτά τα δεδομένα περιλαμβάνουν το session cookie, επιτρέπουν σε αυτόν, να μιμηθεί το θύμα, ακόμα και αν ο κωδικός πρόσβασης δεν είναι εκτεθειμένος. Μη ασφαλή Wi-Fi hotspots είναι ιδιαίτερα ευάλωτα, από την στιγμή που ο οποιοσδήποτε μοιράζεται το δίκτυο μπορεί γενικά να είναι σε θέση να διαβάσει πολλά πράγματα που αφορούν την κυκλοφορία του δικτύου μεταξύ άλλων κόμβων (nodes) και του σημείου πρόσβασης.
- ✓ Εναλλακτικά, ένας εισβολέας με **φυσική πρόσβαση** μπορεί απλώς να επιχειρήσει να κλέψει το session key με το να ανακτήσει, για παράδειγμα, τα περιεχόμενα από το κατάλληλο μέρος του αρχείου ή της μνήμης είτε από τον υπολογιστή του θύματος είτε από το διακομιστή.
- ✓ **Cross-site scripting (XSS)**, όπου ο επιτιθέμενος εξαπατά τον υπολογιστή του χρήστη με το να εκτελέσει JavaScript κώδικα, ο οποίος αντιμετωπίζεται ως αξιόπιστος επειδή

φαίνεται να είναι μέρος της εφαρμογής, επιτρέποντας στον επιτιθέμενο να ανακτήσει ένα αντίγραφο του cookie ή να εκτελέσει άλλες λειτουργίες.

Επιπλέον, ο εισβολέας μπορεί να συνδυάσει την εν λόγω επίθεση με μια επίθεση **ωμής βίας (brute force)**. Αυτό συνεπάγεται ότι θα προσπαθήσει να εισάγει διάφορες session ID τιμές μέχρι να καταφέρει να παρακάμψει τον έλεγχο ταυτότητας των χρηστών.

Μερικές εφαρμογές και εργαλεία που χρησιμοποιούνται για την υλοποίηση αυτής της επίθεσης είναι:

- ✓ Το WhatsApp sniffer
- ✓ Το DroidSheep
- ✓ Το CookieCadger

Οι μέθοδοι που προλαμβάνουν το session hijacking περιλαμβάνουν:

- ✓ Τη κωδικοποίηση των δεδομένων κίνησης (traffic) που περνούν μεταξύ του χρήστη και της εφαρμογής, ιδίως του session key, η οποία θα πρέπει να πραγματοποιείται ιδανικά καθ όλη την διάρκεια κίνησης (των δεδομένων) στη συνολική διάρκεια ζωής της συνόδου με χρήση SSL/TLS. Σε αυτή τη τεχνική βασίζονται ευρέως διαδικτυακές τράπεζες και άλλες ηλεκτρονικές εμπορικές υπηρεσίες, διότι εμποδίζει πλήρως επιθέσεις που έχουν χαρακτήρα sniffing. Ωστόσο, θα μπορούσε να είναι ακόμα δυνατόν να εκτελεστούν μερικά άλλα είδη session hijack. Σε απάντηση, οι επιστήμονες από το Radbound University Nijmegen πρότειναν το 2013 έναν τρόπο πρόληψης session hijacking με το να συσχετίζονται τα session των εφαρμογών με τα SSL/TLS διαπιστευτήρια.
- ✓ Τη χρήση ενός μακρύ τυχαίου αριθμού ή αλφαριθμητικού ως session key. Αυτό μειώνει τον κίνδυνο να μαντέψει ο επιτιθέμενος το έγκυρο session key μέσω δοκιμής και λάθους (error) ή μέσω βίαιων επιθέσεων (brute force attacks).
- ✓ Αναπαράγοντας το session ID μετά από μια επιτυχημένη σύνδεση (login). Αυτό προλαμβάνει μια επίθεση session fixation επειδή ο επιτιθέμενος δε γνωρίζει το session ID του χρήστη.
- ✓ Οι χρήστες μπορούν επίσης να κάνουν αποσύνδεση (log out) από τις ιστοσελίδες κάθε φορά που θέλουν να φύγουν από αυτές. Με την αποσύνδεση, η τιμή του session ID δεν μπορεί να επαναχρησιμοποιηθεί. Ωστόσο, αυτό δεν προστατεύει τους χρήστες από επιθέσεις που υλοποιούνται με χρήση πρόσθετων (extensions), όπως π.χ. με τη χρήση του Firesheep. [36] [37]

2.2.6 Cross-Site Cooking

Το Cross-Site Cooking είναι ένας τύπος εκμετάλλευσης περιηγητή που επιτρέπει σε έναν εισβολέα να ορίσει ένα cookie σε ένα πρόγραμμα περιήγησης μέσα στο cookie domain ενός άλλου εξυπηρετητή. Μπορεί επίσης να χρησιμοποιηθεί για την εκτέλεση μιας επίθεσης **κακόβουλου ορισμού αναγνωριστικών συνόδου (Session Fixation)**, σε περίπτωση που μια κακόβουλη ιστοσελίδα μπορεί να ορίσει το αναγνωριστικό συνόδου (session id) σε ένα cookie, το οποίο ανήκει σε μια άλλη ιστοσελίδα.

Επιπλέον, υπάρχουν περιπτώσεις όπου ο εισβολέας πραγματοποιεί την εν λόγω επίθεση όταν γνωρίζει τα τρωτά σημεία του εξυπηρετητή, τα οποία μπορεί να εκμεταλλευτεί με τη χρήση ενός cookie. Ωστόσο, αν αυτή η ευπάθεια ασφάλειας απαιτεί, για παράδειγμα τον κωδικό πρόσβασης του διαχειριστή και ο εισβολέας δε τον γνωρίζει, η επίθεση αυτή μπορεί να εκτελεστεί από το ίδιο το θύμα αυτόματα.

Η εν λόγω επίθεση μοιάζει πολύ με τις παρακάτω επιθέσεις:

- Cross-site Scripting (XSS)
- Cross-site Request Forgery (CSRF)
- Cross-site Tracing (XST)
- Cross-zone Scripting κλπ.

σε ότι αφορά τη δυνατότητα μεταφοράς δεδομένων ή κώδικα μεταξύ διαφορετικών ιστοσελίδων (ή σε μερικές περιπτώσεις, μεταξύ ανταλλαγής e-mail ή άμεσων μηνυμάτων). Τα προβλήματα αυτά συνδέονται με το γεγονός ότι ένας περιηγητής είναι μια κοινή πλατφόρμα για διαφορετικές πληροφορίες / εφαρμογές / ιστοσελίδες. Μόνο λογικά όρια ασφαλείας εξασφαλίζουν ότι μια ιστοσελίδα δεν μπορεί να καταστρέψει ή να κλέψει δεδομένα που ανήκουν σε μία άλλη. Ωστόσο, ένας τύπος εκμετάλλευσης περιηγητή όπως η επίθεση Cross-site cooking μπορεί να χρησιμοποιηθεί για να αλλάξει τα λογικά όρια ασφαλείας.

Για την αντιμετώπιση αυτής της επίθεσης θα πρέπει οι χρήστες να είναι ενημερωμένοι και να προστατεύουν τόσο τον υπολογιστή τους όσο και το πρόγραμμα περιήγησης που χρησιμοποιούν. [41] [42]

2.2.7 Cross-Site History Manipulation (XSHM)

Αυτή η σχετικά νέα επίθεση βασίζεται στη λειτουργία του περιηγητή που αποθηκεύει το ιστορικό (περιήγησης) του χρήστη. Όταν ο επιτιθέμενος είναι σε θέση να εκμεταλλευτεί το ιστορικό περιήγησης, μπορεί να θέσει σε κίνδυνο την πολιτική ιδιο-προέλευσης (same-origin policy ή SOP) του περιηγητή και να παραβιάσει την ιδιωτικότητα του χρήστη.

Το μοντέλο ασφαλείας SOP χρησιμοποιείται για την επιβολή ορισμένων περιορισμών πρόσβασης στις εφαρμογές διαδικτύου. Το SOP προσδιορίζει ένα πλαίσιο για κάθε σελίδα με βάση την προέλευσή της (origin), η οποία είναι ο μοναδικός συνδυασμός τριών τιμών:

- ✓ του πρωτόκολλου,
- ✓ του τομέα (domain) και
- ✓ της θύρας (port)

Αυτό πρακτικά σημαίνει ότι σελίδες διαφορετικής προέλευσης δεν μπορούν να επικοινωνούν μεταξύ τους (μέσω αποστολής αιτήσεων και λήψης απαντήσεων HTTP).

Μια επίθεση Cross-Site History Manipulation (XSHM) μπορεί να οδηγήσει σε επίθεση **Cross-Site Request Forgery (CSRF)** και σε άλλες εκμεταλλεύσεις, όπως:

- ✓ Σε αναγνώριση κατάστασης σύνδεσης, λάθους κλπ (Cross-site condition leakage)
- ✓ Σε καταγραφή δραστηριότητας των χρηστών (Cross-site user tracking)
- ✓ Σε υποκλοπή παραμέτρων URL (Cross-site URL/parameters enumeration) κ.α.

Για την πρόληψη από αυτή την επίθεση είναι υπεύθυνοι:

- ✓ οι πάροχοι προγραμμάτων περιήγησης, για την σωστή λειτουργία του ιστορικού περιήγησης καθώς και
- ✓ οι προγραμματιστές διαδικτυακών εφαρμογών, σε ότι αφορά τις λειτουργίες ανακατεύθυνσης και τη σωστή διαχείριση των αναγνωριστικών συνόδου (tokens). [38]

2.2.8 Cross-Site Scripting (XSS)

Οι επιθέσεις Cross-Site Scripting (XSS) οφείλονται στην αδυναμία μιας εφαρμογής να ελέγχει τα δεδομένα που εισάγουν οι χρήστες είτε μέσα από τη συμπλήρωση μιας φόρμας είτε από οποιοδήποτε άλλο σημείο εισόδου. Αυτά τα δεδομένα συνήθως περιλαμβάνουν κακόβουλο JavaScript κώδικα ο οποίος εκτελείται στον περιηγητή του θύματος και συμβάλει στη μη εξουσιοδοτημένη πρόσβαση σε ευαίσθητες ή εμπιστευτικές πληροφορίες, στη κατασκοπία πλοήγησης των χρηστών κοκ. Υπάρχουν περιπτώσεις που ο κακόβουλος κώδικας μπορεί να είναι γραμμένος σε VBSScript, ActiveX, Java, Flash ή σε οποιαδήποτε άλλη γλώσσα υποστηρίζεται από μια εφαρμογή πλοήγησης ιστού.

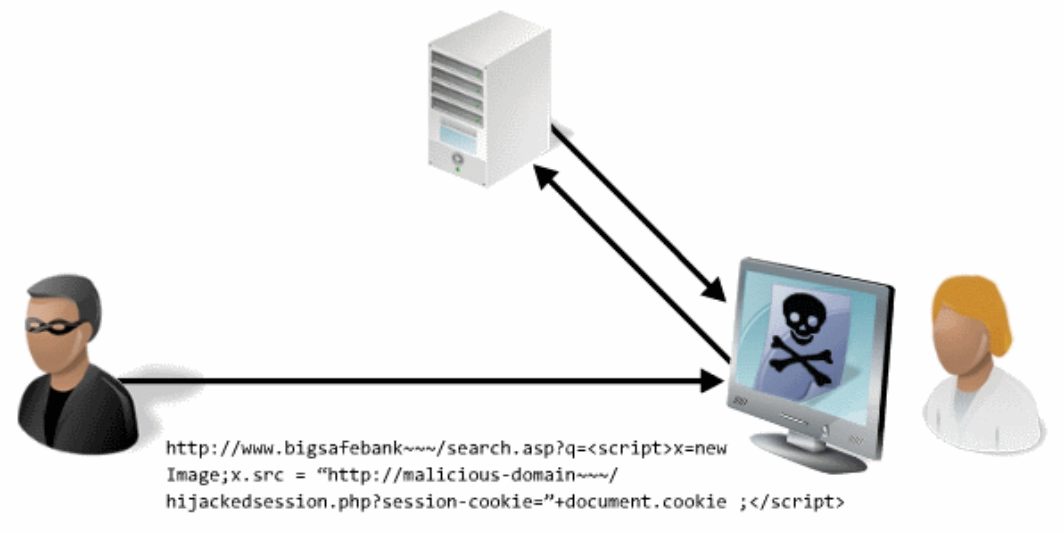
Υπάρχουν δυο ειδών κατηγορίες XSS επιθέσεων:

- οι **μη επίμονες (non-persistent) ή αντανακλώμενες (reflected)** και
- οι **επίμονες (persistent) ή αποθηκευμένες (stored)**.

Μερικές πηγές διαιρούν αυτές τις κατηγορίες παραδοσιακά:

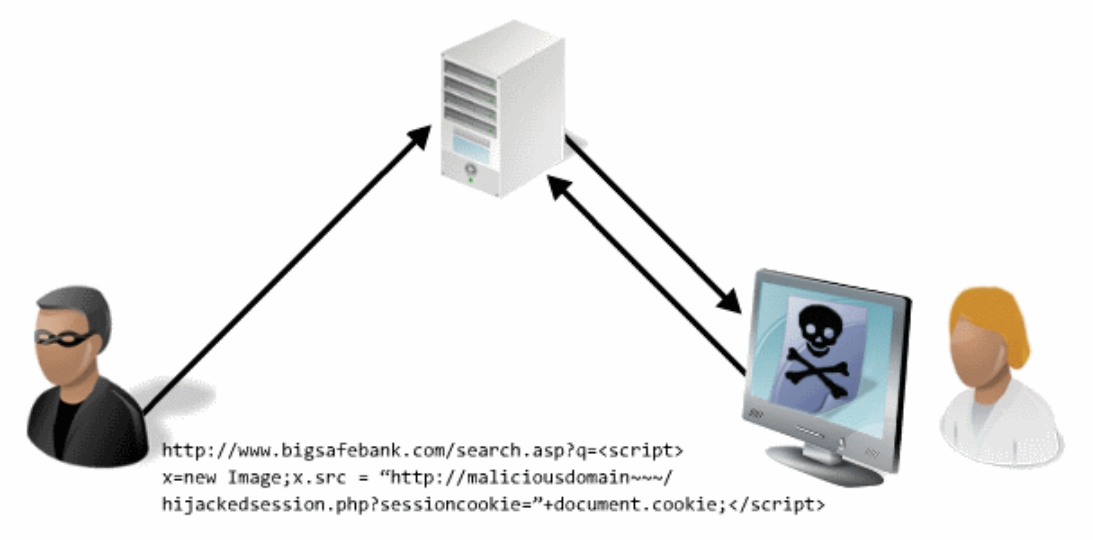
- με βάση τα ελαττώματα από την πλευρά του διακομιστή (server-side) και
- με βάση τα ελαττώματα από την πλευρά του πελάτη (client-side), δηλαδή τα DOM-based.

Υπάρχουν πολλοί τρόποι με τους οποίους ένας εισβολέας μπορεί να οδηγήσει το θύμα σε μια επίθεση **αντανάκλασης (reflective)**. Για παράδειγμα, ο επιτιθέμενος μπορεί να στείλει στο θύμα ένα παραπλανητικό email με έναν σύνδεσμο (link) που να περιέχει κακόβουλο JavaScript κώδικα. Αν το θύμα κάνει κλικ σε αυτό το σύνδεσμο, ένα HTTP αίτημα θα εκτελεστεί από τον περιηγητή του θύματος και θα σταλεί στην ευάλωτη εφαρμογή. Στη συνέχεια, ο κακόβουλος JavaScript κώδικας αντανακλάται πίσω στον περιηγητή του θύματος, όπου και εκτελείται.



Εικόνα 2.2 - Non Persistent XSS Attack [44]

Οι **επίμονες (persistent)** XSS επιθέσεις είναι μια κατηγορία που περιλαμβάνει πιο καταστροφικά ελαττώματα. Τέτοιες επιθέσεις παρατηρούνται όταν τα δεδομένα που παρέχονται από τον επιτιθέμενο έχουν αποθηκευτεί στον εξυπηρετητή, και στη συνεχεία εμφανίζονται μόνιμα σε "κανονικές" σελίδες που βλέπουν οι χρήστες κατά την περιήγησή τους.



Εικόνα 2.3 - Persistent XSS Attack [44]

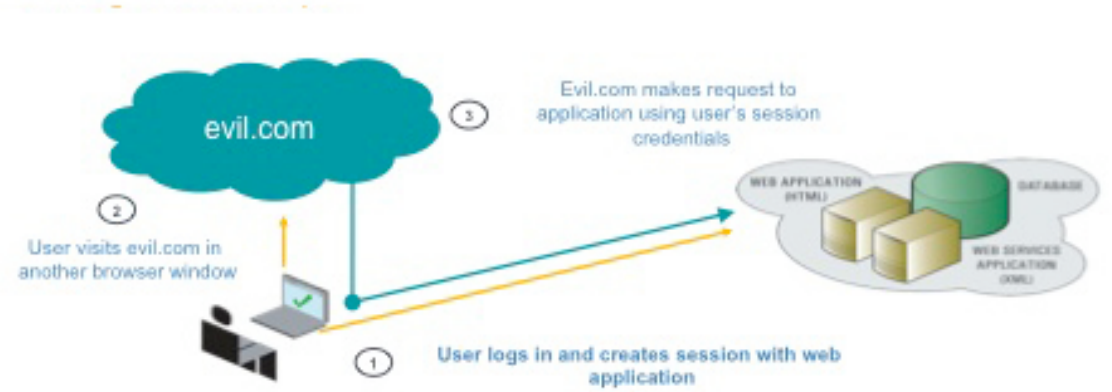
Οι **επίμονες (persistent)** XSS επιθέσεις είναι πιο επικίνδυνες από άλλους τύπους επειδή το κακόβουλο script καθίσταται αυτόματα χωρίς να χρειαστεί να στοχεύσει σε μεμονωμένα θύματα ή να χρειαστεί να δελεαστούν τα θύματα μέσω κάποιας άλλης ιστοσελίδας. Ιδιαίτερα στην περίπτωση των ιστοσελίδων κοινωνικής δικτύωσης, ο κώδικας συνήθως σχεδιάζεται με τέτοιο τρόπο έτσι ώστε να αυτό-διαδίδεται, δημιουργώντας ένα είδος **client-side worm**.

Για την αντιμετώπιση από αυτή την επίθεση θα πρέπει:

- ✓ Να εφαρμοστούν τα μέτρα προστασίας για την ευπάθεια **του ακατάλληλου χειρισμού εισόδου & εξόδου (Improper Input & Output Handling)** των δεδομένων που εισάγουν οι χρήστες.
- ✓ Να χρησιμοποιούνται υπηρεσίες σάρωσης, οι οποίες προσομοιώνουν μια επίθεση από τον εξυπηρετητή μιας εταιρείας προς τον εξυπηρετητή της εφαρμογής του πελάτη. Αν η επίθεση πραγματοποιηθεί επιτυχώς, ο πελάτης θα λάβει αναλυτικές πληροφορίες σχετικά με τον τρόπο που πραγματοποιήθηκε η επίθεση προκειμένου να είναι σε θέση να αμυνθεί πριν δεχτεί ξανά την ίδια επίθεση από κάποιον κακόβουλο χρήστη.
- ✓ Να πραγματοποιούνται χειροκίνητοι έλεγχοι οι οποίοι πολλές φορές είναι σε θέση να εντοπίσουν ευπάθειες που οι υπηρεσίες σάρωσης αδυνατούν να εντοπίσουν.
- ✓ Να εγκαταστήσουν οι χρήστες κάποιο plug-in ή να ρυθμίσουν το περιηγητή τους έτσι ώστε να μπλοκάρουν τα scripts που εκτελούνται από την πλευρά τού (client-side) με βάση το όνομα της ιστοσελίδας (domain name). Πάνω σε αυτό θα πρέπει επίσης να αναφερθεί ότι ανάλογα με το περιηγητή που χρησιμοποιεί ένας χρήστης υπάρχει και το κατάλληλο πρόσθετο (add-on) που παρέχει προστασία κατά των XSS επιθέσεων, ακόμα και στην περίπτωση που τα script είναι ενεργοποιημένα για όλες τις ιστοσελίδες.
- ✓ Να μην αποθηκεύουν οι χρήστες τα συνθηματικά τους (usernames & passwords) στον περιηγητή τους επειδή υπάρχει κίνδυνος υποκλοπής. [43] [44] [45] [46] [47]

2.2.9 Cross-Site Request Forgery (CSRF)

Η Cross-Site Request Forgery (CSRF ή XSRF) είναι μια κοινή και σοβαρή επίθεση όπου ο χρήστης εξαπατάται και προβαίνει σε ενέργειες που ο ίδιος δεν θέλησε ρητά να κάνει. Κάτι τέτοιο μπορεί να συμβεί με την αποστολή ενός HTTP αιτήματος από μία κακόβουλη ιστοσελίδα προς την ιστοσελίδα-στόχο μέσα από τον περιηγητή του θύματος. Σε αντίθεση με μια επίθεση XSS, η οποία εκμεταλλεύεται την εμπιστοσύνη που δείχνει ο χρήστης προς την ιστοσελίδα-στόχο, η CSRF εκμεταλλεύεται την εμπιστοσύνη που έχει η ιστοσελίδα-στόχος προς το χρήστη.



Εικόνα 2.4 Cross-Site Request Forgery (CSRF) [152]

Για να πετύχει μια CSRF επίθεση, θα πρέπει ο επιτιθέμενος να παρασύρει το θύμα σε μια ιστοσελίδα με κακόβουλο περιεχόμενο ενώ το θύμα είναι ήδη συνδεδεμένο (logged-in) στην ιστοσελίδα-στόχο. Η επίθεση είναι "τυφλή", πράγμα που σημαίνει ότι ο επιτιθέμενος δεν μπορεί να δει τι στέλνει η ιστοσελίδα-στόχος πίσω στο θύμα σε απάντηση των πλαστών HTTP αιτήσεων, εκτός αν εκμεταλλεύεται μια XSS ευπάθεια ή κάποιο άλλο προγραμματιστικό σφάλμα (bug) στην ιστοσελίδα-στόχο. Μια επίθεση CSRF είναι επίσης γνωστή σαν **one-click attack** , **session riding** και **Hostile linking**.

Εκτός από την κλασική επίθεση CSRF υπάρχει και η επίθεση **Login CSRF**. Πρόκειται για μια παραλλαγή της κλασικής επίθεσης CSRF με τη διαφορά ότι ο επιτιθέμενος στέλνει ένα πλαστό HTTP αίτημα πριν συνδεθεί το θύμα στην ιστοσελίδα-στόχο. Το αίτημα αυτό έχει σαν στόχο να συνδέσει (login) το θύμα με τα διαπιστευτήρια του εισβολέα.

Πέρα από τους παραπάνω στατικούς τύπους επιθέσεων, μπορούν να κατασκευαστούν και δυναμικές επιθέσεις, ως μέρος ενός ωφέλιμου φορτίου για μια επίθεση XSS, με την μορφή ενός client-side worm (όπως π.χ. το Samy worm) ή από πληροφορίες συνεδριών (sessions) που

έχουν διαρρεύσει μέσω περιεχομένου που βρίσκεται εκτός της ιστοσελίδας-στόχου και στέλνονται στην ιστοσελίδα του επιτιθέμενου μέσα από ένα κακόβουλο URL. Τα CSRF tokens μπορούν επίσης να σταλούν σε έναν χρήστη (client) από έναν επιτιθέμενο κατά την διάρκεια μιας **session fixation** ή μέσω μιας επίθεσης **ωμής βίας (brute-force)**, κατά την οποία παράγονται χιλιάδες αποτυχημένες αιτήσεις.

Οι περισσότερες τεχνικές πρόληψης λειτουργούν με την ενσωμάτωση πρόσθετων στοιχείων πιστοποίησης μέσα στα αιτήματα που επιτρέπουν στις διαδικτυακές εφαρμογές να ανιχνεύσουν αιτήματα από μη εξουσιοδοτημένες τοποθεσίες.

Το **Synchronizer Token Pattern** περιλαμβάνει μια τεχνική όπου το διακριτικό (token), που είναι μυστικό και μοναδικό για κάθε αίτημα, είναι ενσωματωμένο σε όλες τις HTML φόρμες της εφαρμογής και επαληθεύεται από την πλευρά του διακομιστή. Το token μπορεί να παραχθεί με οποιαδήποτε μέθοδο η οποία εξασφαλίζει ότι θα είναι απρόβλεπτο και μοναδικό (π.χ. με χρήση τυχαίων hash τιμών) και συνδέεται με την τρέχουσα σύνοδο πριν ενσωματωθεί σε μια φόρμα. Κατά συνέπεια, ο επιτιθέμενος δεν μπορεί να τοποθετήσει ένα σωστό token στα αιτήματά του προκειμένου να τα πιστοποιήσει.

Μερικές ιστοσελίδες επιβάλουν στους χρήστες να εισάγουν ξανά τα διαπιστευτήρια τους πριν από την εκτέλεση κρίσιμων λειτουργιών, όπως π.χ. πριν τη μεταφορά χρημάτων ή την αλλαγή κωδικού πρόσβασης. Σε πιο απλές περιπτώσεις, όπως κατά την υποβολή ενός μηνύματος, χρησιμοποιούνται μηχανισμοί CAPTCHA όπως το Google reCaptcha. Τέτοιοι **επιπρόσθετοι μηχανισμοί πιστοποίησης** είναι αποτελεσματικοί ενάντια σε CSRF επιθέσεις, αλλά απαιτούν επιπλέον αλληλεπίδραση με το χρήστη.

Εκτός από τις παραπάνω μεθόδους προστασίας υπάρχουν αρκετές ακόμα που μπορεί να χρησιμοποιήσει ένας προγραμματιστής, χωρίς αυτό να σημαίνει ότι είναι εξίσου αποτελεσματικές. [48] [49] [50] [51]

Από την άλλη πλευρά, οι χρήστες (clients) διαδικτυακών εφαρμογών οφείλουν:

- ✓ Να κάνουν αποσύνδεση (log-out) όταν δεν χρησιμοποιούν την εφαρμογή.
- ✓ Να μην αποθηκεύουν τα διαπιστευτήρια τους (username / password) στον περιηγητή τους.
- ✓ Να χρησιμοποιούν έναν περιηγητή για να επισκέπτονται σημαντικές ιστοσελίδες και έναν άλλο για να επισκέπτονται όλες τις υπόλοιπες.
- ✓ Να έχουν εγκατεστημένα πρόσθετα (plug-ins) στον περιηγητή τους. Πρόσθετα όπως το No-Script κάνουν ακόμα πιο δύσκολη την υλοποίηση μια επίθεσης CSRF.

2.2.10 Cross-Site Tracing

Μια Cross-Site Tracing (XST) επίθεση εκμεταλλεύεται ένα ActiveX, Flash, Java ή κάποιο άλλο στοιχείο ελέγχου που επιτρέπει την εκτέλεση μιας HTTP TRACE αίτησης (request). Η επίθεση αυτή, ανακαλύφθηκε από τον ερευνητή διαδικτυακής ασφάλειας Jeremiah Grossman το 2003, και επιτρέπει σε έναν εισβολέα να αποκτήσει πρόσβαση σε cookies και διαπιστευτήρια πιστοποίησης ενός χρήστη.

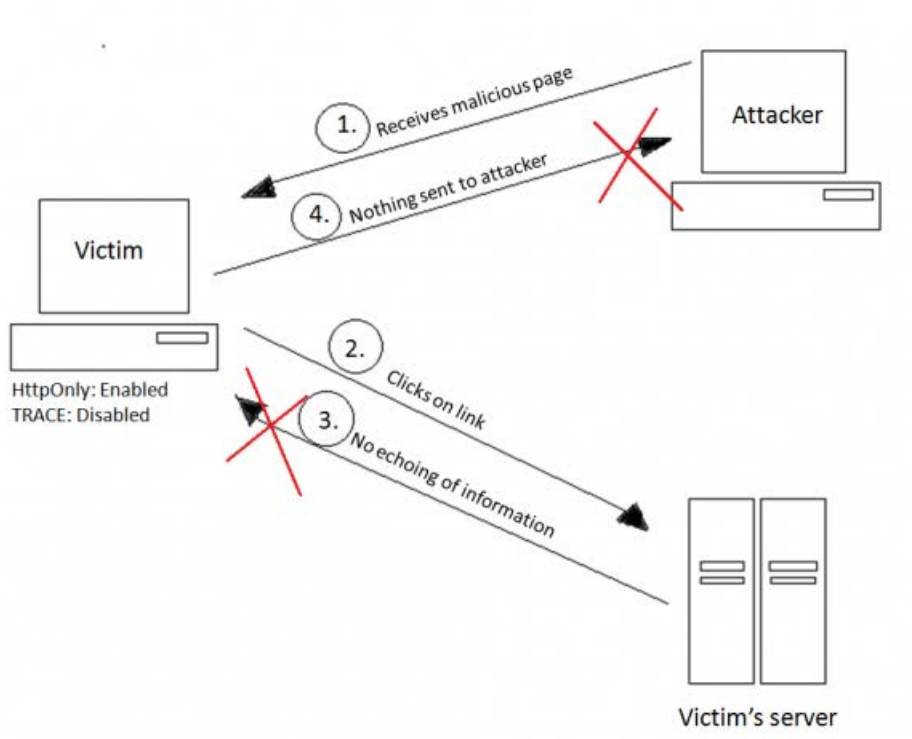
Μια XST επίθεση συνδυάζει μια **Cross-Site Scripting (XSS)** επίθεση, όπου ο εισβολέας εισάγει κακόβουλο κώδικα σε κάποιο σύνδεσμο, με μια HTTP TRACE μέθοδο. Οι περισσότεροι προγραμματιστές διαδικτύου είναι εξοικειωμένοι με τις HTTP μεθόδους GET και POST, οι οποίες στέλνουν αιτήσεις προς τον εξυπηρετητή και λαμβάνουν πληροφορίες πρόσβασης από αυτόν, αλλά υπάρχουν και αρκετές άλλες που είναι λιγότερο γνωστές μέθοδοι, συμπεριλαμβανομένης και της TRACE μεθόδου.

Η HTTP TRACE ζητά από έναν εξυπηρετητή να εμφανίσει τα περιεχόμενα μιας αίτησης πίσω στο χρήστη (client). Η ολοκληρωμένη αίτηση, συμπεριλαμβάνει τις HTTP επικεφαλίδες, οι οποίες μπορούν να περιλαμβάνουν ευαίσθητες πληροφορίες, όπως cookies και στοιχεία ταυτότητας, επιστρέφεται στο σώμα (entity-body) μιας TRACE απάντησης. Η αίτηση αυτή χρησιμοποιείται κυρίως από τους προγραμματιστές για να πραγματοποιήσουν ελέγχους και να διορθώσουν προγραμματιστικά λάθη (debugging) σε HTTP εφαρμογές και είναι διαθέσιμη από προεπιλογή στο λογισμικό των περισσότερων εξυπηρετητών διαδικτύου.

Ένας τρόπος για να υλοποιηθεί μια XST επίθεση είναι να δημιουργήσει ένας κακόβουλος χρήστης μια ιστοσελίδα που να περιλαμβάνει ένα TRACE αίτημα σε JavaScript. Η JavaScript μπορεί στη συνέχεια να εκμεταλλευτεί κάθε ευπάθεια μεταξύ διάφορων ονομάτων χώρου, συμπεριλαμβανομένων και εκείνων που χρησιμοποιούν SSL, με σκοπό να συλλέξει από τον περιηγητή ενός επισκέπτη τα στοιχεία ταυτότητας που είναι προσωρινά αποθηκευμένα.

Μια άλλη ακόμα πιο συνηθισμένη μέθοδος χρησιμοποιεί ένα JavaScript snippet που περιλαμβάνει μια TRACE αίτηση και την εισάγει μέσα σε μια ευπαθή διαδικτυακή εφαρμογή. Η JavaScript θα είναι σε θέση να στείλει τις επικεφαλίδες των αιτημάτων του θύματος, συμπεριλαμβανομένων των cookie δεδομένων τα οποία είναι χαρακτηρισμένα ως "httpOnly", στον εισβολέα. Η "httpOnly" είναι μια επιπλέον παράμετρος που προστίθεται στα cookies, η οποία τα κράτα κρυφά από τα scripts και υποστηρίζεται από τους περισσότερους περιηγητές, ωστόσο η TRACE μέθοδος μπορεί να χρησιμοποιηθεί για να παρακάμψει αυτή την προστασία. Είναι εύκολο για τον εισβολέα, ή για το διαχειριστή του συστήματος, να ελέγξει αν ο εξυπηρετητής υποστηρίζει την TRACE μέθοδο. Κάνοντας χρήση κάποιου βοηθητικού προγράμματος, όπως π.χ. του Netcat, που μπορεί να διαβάσει τις συνδέσεις δικτύου με πρωτόκολλο TCP ή UDP, ο επιτιθέμενος μπορεί να χρησιμοποιήσει τις επιλογές που αφορούν τις HTTP μεθόδους προκειμένου να ανακτήσει μια λίστα με τις μεθόδους που υποστηρίζει ο εξυπηρετητής.

Για την πρόληψη αυτής της επίθεσης, είναι απαραίτητο να μπλοκαριστούν από τον εξυπηρετητή οι PUT, DELETE, CONNECT και TRACE μέθοδοι αφού μπορούν να προκαλέσουν προβλήματα ασφάλειας.



Εικόνα 2.5 - Cross-site Tracing [53]

Αν μια εφαρμογή χρειάζεται μια ή περισσότερες από αυτές τις μεθόδους, είναι σημαντικό να ελέγχετε η χρήση τους, ενώ πρέπει να χρησιμοποιούνται υπό ασφαλείς συνθήκες και από έμπιστους χρήστες. Για να απενεργοποιηθεί η HTTP TRACE σε Apache εξυπηρετητή, θα πρέπει να οριστεί η TraceEnable σε κατάσταση 'off'. Σε IIS και σε Windows εξυπηρετητή, χρησιμοποιείται το URLScan εργαλείο για να απορρίπτονται HTTP TRACE αιτήσεις ή για να επιτρέπονται μόνο οι μέθοδοι που είναι απαραίτητες και δεν επηρεάζουν την πολιτική ασφαλείας της εφαρμογής. [52] [53]

2.2.11 Cross-user Defacement

Ένας εισβολέας μπορεί να στείλει ένα απλό αίτημα σε έναν ευάλωτο εξυπηρετητή έτσι ώστε να τον αναγκάσει να δημιουργήσει δυο απαντήσεις, εκ των οποίων η δεύτερη μπορεί να παρερμηνευθεί ως απάντηση σε ένα διαφορετικό αίτημα. Η επίθεση αυτή μπορεί να επιτευχθεί είτε με το να αναγκαστεί ο χρήστης να υποβάλει ο ίδιος το αίτημα, από μόνος του, είτε εξ αποστάσεως από τον εισβολέα, αν χρήστης και εισβολέας μοιράζονται την ίδια TCP σύνδεση με τον εξυπηρετητή, όπως όταν μοιράζονται τον ίδιο ενδιάμεσο εξυπηρετητή (proxy).

Στην καλύτερη περίπτωση, ένας εισβολέας μπορεί να αξιοποιήσει την επίθεση αυτή προκειμένου να κάνει τους χρήστες να χάσουν την εμπιστοσύνη τους ως προς την ασφάλεια της εφαρμογής. Στην χειρότερη περίπτωση, ένας εισβολέας μπορεί να παρουσιάσει ένα ειδικά διαμορφωμένο περιεχόμενο το οποίο είναι σχεδιασμένο για να μιμείται τη συμπεριφορά της εφαρμογής, προκειμένου να ανακατευθύνει πίσω στον εισβολέα τις προσωπικές πληροφορίες των χρηστών, όπως αριθμούς λογαριασμών και κωδικούς πρόσβασης.

Η επίθεση αυτή βασίζεται σε σφάλματα που υπάρχουν στην εφαρμογή και σε επιθέσεις **διάσπασης απαντήσεων HTTP (HTTP Response Splitting)**. Είναι ζωτικής σημασίας από την πλευρά του εισβολέα το ότι η εφαρμογή επιτρέπει στο πεδίο συμπλήρωσης επικεφαλίδας (header) την εισαγωγή περισσότερων από μία επικεφαλίδες με χρήση CR (Carriage Return) και LF (Line Feed) χαρακτήρων.

Επειδή, η εν λόγω επίθεση είναι αποτέλεσμα της επίθεσης **διάσπασης απαντήσεων HTTP (HTTP Response Splitting)**. Αυτό σημαίνει ότι ένας προγραμματιστής διαδικτυακών εφαρμογών οφείλει να είναι ιδιαίτερα προσεκτικός κατά το στάδιο υλοποίησης της εφαρμογής και να εφαρμόσει τα αντίμετρα που απαιτούνται για την πρόληψη της επίθεσης διάσπασης αιτήσεων HTTP. [56] [57]

2.2.12 Cross-zone scripting

Αυτή η επίθεση επιτρέπει σε έναν εισβολέα να προκαλέσει ένα θύμα να φορτώσει κακόβουλο περιεχόμενο μέσα στον περιηγητή του. Το περιεχόμενο αυτό, μπορεί να είναι κάποιο script κώδικα ή άλλα διαδικτυακά αντικείμενα όπως μη υπογεγραμμένα ActiveX στοιχεία ελέγχου ή διάφορα applets, έχει σαν στόχο να παρακάμπτει τους ελέγχους ζώνης ασφαλείας του περιηγητή του θύματος. Αυτή η επίθεση μπορεί να επιτευχθεί με την εκμετάλλευση σφαλμάτων (bugs) στον περιηγητή, με την εκμετάλλευση εσφαλμένης διαμόρφωσης στους έλεγχους ζώνης (του περιηγητή) αλλά και μέσα από μια επίθεση Cross-site Scripting (XSS).

Το να γίνεται επαρκής επικύρωση των δεδομένων που εισάγει ο χρήστης , να εξασφαλιστεί η σωστή κωδικοποίηση του HTML κώδικα που εξάγεται πριν από την εγγραφή των δεδομένων του χρήστη που παρέχονται στην σελίδα και να απενεργοποιείται η εκτέλεση των script, όπου είναι απαραίτητο, είναι μερικές από τις ενέργειες που μπορεί να κάνει ένας προγραμματιστής προκειμένου να προστατέψει την εφαρμογή του. [54] [55]

2.2.13 **Άρνηση υπηρεσίας** (Denial of Service, DoS)

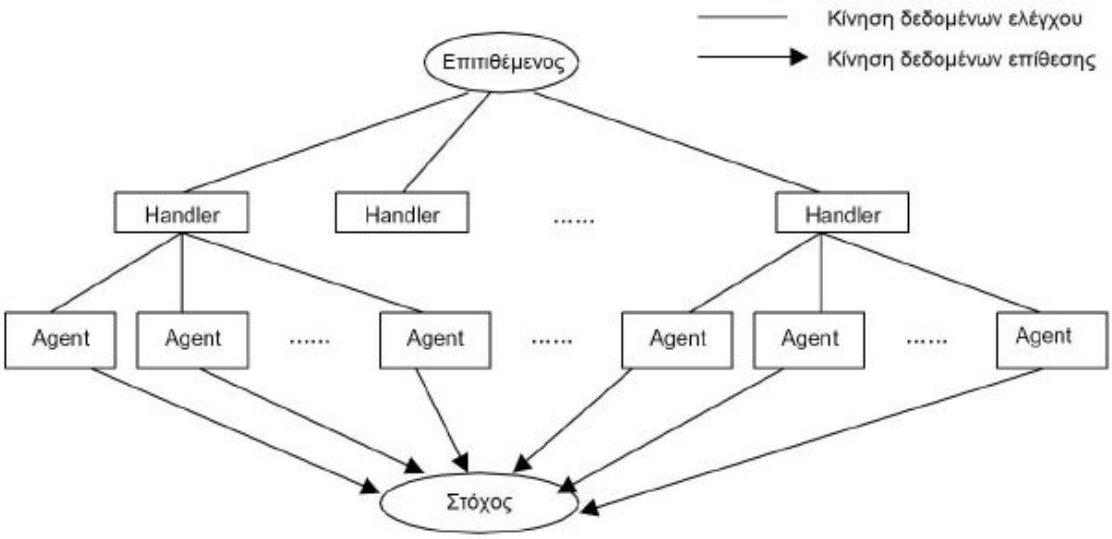
Οι επιθέσεις άρνησης υπηρεσίας είναι μια μεγάλη κατηγορία επιθέσεων, εκ των οποίων κάθε μια από αυτές θέτουν μια εφαρμογή εκτός λειτουργίας. Ο εισβολέας έχει σαν στόχο να εκμεταλλευτεί τα τρωτά σημεία που υπάρχουν στο δίκτυο, στην εφαρμογή, στο εξυπηρετητή ή σε οποιοδήποτε άλλο επιμέρους σύστημα (π.χ. σε εξυπηρετητές DNS, σε εξυπηρετητές βάσης δεδομένων κ.λπ.) χρησιμοποιώντας την κατάλληλη μέθοδο.

Γενικά υπάρχουν δυο γενικές κατηγορίες που οδηγούν σε άρνηση υπηρεσίας. Η πρώτη κατηγορία είναι γνωστή ως **"πλημμύρα" (flood)** και περιλαμβάνει επιθέσεις κατά τις οποίες η εφαρμογή δέχεται έναν μεγάλο αριθμό ψευδών αιτήσεων (requests) για εξυπηρέτηση. Αυτό έχει σαν αποτέλεσμα να σταματήσει η εφαρμογή να είναι διαθέσιμη στους χρήστες, αφού δεν είναι δυνατόν να αποσταλεί ο αντίστοιχος αριθμός απαντήσεων (responses). Η δεύτερη κατηγορία περιλαμβάνει επιθέσεις κατά τις οποίες η εφαρμογή **"καταρρέει" (crash)**. Κατά τη διάρκεια επιθέσεων "κατάρρευσης", ο εισβολέας μπορεί να θέσει μια εφαρμογή εκτός λειτουργίας με το

να εγχύσει και να εκτελέσει κακόβουλο κώδικα μέσα στην εφαρμογή ή με το να κάνει **κατάχρηση της λειτουργικότητάς (Abuse of functionality)** της. Οι επιθέσεις "κατάρρευσης" πολλές φορές προσπαθούν να καταναλώσουν όλους τους διαθέσιμους πόρους, όπως π.χ. του επεξεργαστή, της μνήμης, του χώρου στο δίσκο κ.λπ., σε κάποιο από τα επιμέρους συστήματα.

Μια πιο εξελιγμένη μορφή των επιθέσεων άρνησης υπηρεσίας (DoS) είναι οι **Κατανεμημένες Επιθέσεις Άρνησης Υπηρεσίας (Distributed Denial of Service - DDoS)**. Σε μια DDoS επίθεση, ένας εισβολέας μπορεί να χρησιμοποιήσει τον υπολογιστή ενός χρήστη για να επιτεθεί σε ένα σύστημα, χωρίς αυτό να σημαίνει ότι έχει και τη συγκατάθεσή του. Ουσιαστικά ο εισβολέας στέλνει από τον υπολογιστή του θύματος σε μια εφαρμογή τεράστιες ποσότητες δεδομένων ή ανεπιθύμητα μηνύματα αλληλογραφίας σε συγκεκριμένες email διευθύνσεις. Αυτό όμως που πραγματικά κάνει δυνατή την επίθεση είναι ότι ο εισβολέας μπορεί να ελέγχει ταυτόχρονα περισσότερους από έναν υπολογιστές, δημιουργώντας έτσι ένα πανίσχυρο δίκτυο επιτιθέμενων μηχανών, το οποίο μπορεί να υπερφορτώσει ακόμα και έναν ισχυρό σε πόρους εξυπηρετητή. Οι υπολογιστές που βρίσκονται υπό τον έλεγχο του επιτιθέμενου χωρίζονται σε δύο κατηγορίες:

- Στους **Handlers ή Masters**, που είναι οι υπολογιστές στους οποίους εκτελείται ένα ειδικό πρόγραμμα, ικανό να ελέγχει πολλαπλούς πράκτορες (agents) και
- Στους **πράκτορες (agents) ή εξυπηρετητές zombie** που είναι υπολογιστές στους οποίους εκτελείται ένα ειδικό πρόγραμμα και είναι υπεύθυνοι για τη δημιουργία ροής πακέτων που προορίζονται για το θύμα. Οι υπολογιστές αυτοί είναι συνήθως εκτός του δικτύου του θύματος, για να αποφευχθεί η αποδοτική αντίδραση από το θύμα, και εκτός του δικτύου του επιτιθέμενου, για να αποφευχθεί η νομική του ευθύνη αν γίνει προσπάθεια για τον εντοπισμό του.



Εικόνα 2.6 - Distributed Denial of Service - DDoS [66]

Τα εργαλεία που χρησιμοποιούνται σε DDoS επιθέσεις χωρίζονται σε δύο κατηγορίες. Η πρώτη κατηγορία περιλαμβάνει αυτά που βασίζονται στους πράκτορες (agents) και είναι τα εξής:

- Το Trinoo
- Το Tribe Flood Network
- Το TFN2K
- Το Stacheldraht
- Το mstream
- Το Shaft

Η δεύτερη κατηγορία περιλαμβάνει εργαλεία που βασίζονται στο σύστημα ανταλλαγής ηλεκτρονικών μηνυμάτων μεταξύ πολλών χρηστών (Internet Relay Chat - IRC). Μερικά από αυτά είναι:

- Το Trinity v3
- Το Knight

Κατά καιρούς έχουν υλοποιηθεί πολλές επιθέσεις άρνησης υπηρεσίας και παρόλα αυτά συχνά εφευρίσκονται νέες. Παρακάτω αναφέρονται συνοπτικά μερικές από τις πιο γνωστές επιθέσεις που έχουν παρουσιαστεί.

Μερικές επιθέσεις βασίζονται στο πρωτόκολλο ICMP (Internet Control Message), το οποίο είναι ένα από τα βασικά πρωτόκολλα του διαδικτύου. Οι επιθέσεις αυτές αποσκοπούν στην αποστολή κατεστραμμένων ICMP πακέτων (packets) προς το σύστημα-στόχο. Μερικές από τις πιο γνωστές είναι:

- Η επίθεση **Smurf**
- Η επίθεση **Ping flood**
- Η επίθεση **Ping of Death**
- Η επίθεση **Nuke**

Η επίθεση **Fraggle** είναι μια παραλλαγή της επίθεσης smurf με την διαφορά ότι η πρώτη βασίζεται στο πρωτόκολλο UDP (User Datagram Protocol). Το ευχάριστο είναι ότι τα περισσότερα δίκτυα μπορούν να προστατευτούν και από τις δύο επιθέσεις.

Σύμφωνα με το πρωτόκολλο TCP (Transmission Control Protocol) για να δημιουργηθεί μια σύνδεση μεταξύ ενός πελάτη (client) και ενός εξυπηρετητή (server) λαμβάνει χώρα μια ακολουθία τριών βημάτων, γνωστή και ως "τριμερής χειραψία" (three-way handshake). Σε μια επίθεση **SYN Flood** ο επιτιθέμενος έχει σαν στόχο την αποστολή κακόβουλων TCP πακέτων που δεν επιτρέπουν την ολοκλήρωση της εν λόγω ακολουθίας με αποτέλεσμα να καταναλώνονται οι πόροι του εξυπηρετητή μέχρι να τον οδηγήσει σε κατάρρευση. Η πιο αποτελεσματική μέθοδος αντιμετώπισης αυτού του κινδύνου είναι η καταγραφή του αριθμού των συνδέσεων που έχει ξεκινήσει κάθε πελάτης (client) και η απαγόρευση δημιουργίας νέων συνδέσεων όταν ο αριθμός αυτός ξεπεράσει κάποιο προκαθορισμένο όριο.

Σε μία επίθεση **Teardrop** ο επιτιθέμενος εκμεταλλεύεται τις αδυναμίες στην ανασυγκρότηση των IP πακέτων. Όταν ένα IP πακέτο αποστέλλεται στο διαδίκτυο, ενδέχεται να ταξιδεύει τεμαχισμένο σε επιμέρους, μικρότερα τμήματα (fragments). Κάθε τμήμα περιλαμβάνει στην κεφαλή του ένα πεδίο (field), όπου εκεί περιγράφεται η θέση του στο αρχικό, "μεγάλο" πακέτο IP. Ο επιτιθέμενος χρησιμοποιεί ένα πρόγραμμα, που λέγεται "Teardrop", το οποίο τεμαχίζει IP πακέτα σε τμήματα με λανθασμένες πληροφορίες στο πεδίο αυτό. Όταν το σύστημα-στόχος προσπαθήσει να συναρμολογήσει τα "παραπλανητικά" αυτά τμήματα, θα κολλήσει ή θα επανεκκινήσει, εκτός αν ο διαχειριστής του συστήματος έχει φροντίσει να αναβαθμίσει το λειτουργικό με το κατάλληλο patch που διορθώνει το πρόβλημα.

Σε μια επίθεση **IP (Address) Spoofing**, ο εισβολέας στέλνει κακόβουλα πακέτα IP από μια πλαστή IP διεύθυνση, προκειμένου να κρύψει τα ίχνη του. Σκοπός της επίθεσης είναι να υπερφορτωθεί το δίκτυο ή η συσκευή με πακέτα που φαινομενικά προέρχονται από νόμιμες IP διευθύνσεις. Υπάρχουν δύο τρόποι με τους οποίους μια επίθεση IP Spoofing μπορεί να υπερφορτώσει ένα σύστημα-στόχο. Ο ένας τρόπος είναι να πλημμυρίσει το σύστημα-στόχος με πακέτα από πολλαπλές πλαστές IP διευθύνσεις. Αυτός ο τρόπος λειτουργεί με την άμεση αποστολή περισσότερων δεδομένων από αυτά που μπορεί να διαχειριστεί το σύστημα-στόχος. Ο δεύτερος τρόπος στηρίζεται στην εκμετάλλευση της IP διεύθυνσης του συστήματος-στόχου προκειμένου να αποσταλούν πλαστογραφημένα πακέτα από αυτή τη διεύθυνση προς πολλούς διαφορετικούς αποδέκτες στο δίκτυο. Όταν οι αποδέκτες λάβουν τα πακέτα, θα στείλουν αυτόματα αντίστοιχα πακέτα ως απάντηση. Δεδομένου ότι αυτά τα πακέτα φαίνονται να έχουν σταλεί από την IP διεύθυνση του συστήματος-στόχου, όλες οι απαντήσεις που θα αποσταλούν θα "πλημμυρίσουν" την IP διεύθυνση (του συστήματος-στόχου) με πλαστά πακέτα. Πολλές φορές για λόγους ασφαλείας χρησιμοποιούνται αντίστοιχα φίλτρα τα οποία μπλοκάρουν πακέτα με αντικρουόμενες πληροφορίες. Αυτά τα πακέτα συνήθως προέρχονται εκτός του δικτύου αλλά φέρουν τις ίδιες πληροφορίες με αυτές που έχουν τα πακέτα τα οποία προέρχονται από το εσωτερικό του δικτύου και αντιστρόφως.

Το πρωτόκολλο ARP (Address Resolution Protocol) χρησιμοποιείται για να αναλύει τις IP διευθύνσεις σε MAC (Media Access Control) διευθύνσεις για τη μεταφορά δεδομένων. Σε μια επίθεση **ARP Spoofing**, ένας κακόβουλος χρήστης στέλνει πλαστογραφημένα ARP μηνύματα σε ένα τοπικό δίκτυο (Local Area Networks - LAN), προκειμένου να συνδέσει την MAC διεύθυνσή του με την IP διεύθυνση ενός νόμιμου μέλους του δικτύου. Αυτό έχει σαν αποτέλεσμα να στέλνονται τα δεδομένα στον εισβολέα, ο οποίος μπορεί να τα τροποποιήσει, να τα κλέψει καθώς επίσης και να σταματήσει την κυκλοφορία του τοπικού δικτύου. Αυτή η επίθεση λειτουργεί μόνο σε τοπικά δίκτυα που χρησιμοποιούν το πρωτόκολλο ARP. Επιπροσθέτως, η εν λόγω επίθεση μπορεί να ευνοήσει επιθέσεις **Session hijacking** και **Man-in-the-middle (MITM)**.

Το δυσάρεστο με τις περισσότερες επιθέσεις άρνησης υπηρεσίας είναι ότι δεν μπορούν να προβλεφθούν εύκολα. Ωστόσο, μπορούν να εφαρμοστούν μερικά αντίμετρα προκειμένου να αποφευχθούν ως ένα βαθμό. Εκτός από τα αντίμετρα που αναφέρθηκαν παραπάνω για την προστασία ενός συστήματος, απαιτείται:

- ✓ Τακτικός έλεγχος και εγκατάσταση των πιο πρόσφατων ενημερώσεων στο λειτουργικό που χρησιμοποιεί ο εξυπηρετητής. Επιπλέον, θα πρέπει να γίνεται εγκατάσταση των αντίστοιχων patches στο δρομολογητή (router) και στο τείχος προστασίας (firewall).
- ✓ Η συνεχής καταγραφή της κυκλοφορίας του δικτύου προκειμένου να απαγορευτεί η μεταφορά κακόβουλων πακέτων (packets).
- ✓ Η απενεργοποίηση όλων των μη αναγκαίων TCP/ UDP υπηρεσιών.
- ✓ Το τείχος προστασίας (firewall) και ο δρομολογητής (router) που χρησιμοποιούνται είναι ρυθμισμένα έτσι ώστε να μπλοκάρουν το διπλασιασμό της κυκλοφορίας (traffic) στο δίκτυο.
- ✓ Η χρήση σαρωτών για των εντοπισμό ευπαθειών που οδηγούν στην υλοποίηση αυτών των επιθέσεων. Δύο γνωστοί σαρωτές είναι ο QualysQuard και ο WebInspect.

Άξιο αναφοράς είναι το γεγονός ότι οι επιτιθέμενοι ανακαλύπτουν διαρκώς άλλες αδυναμίες των πρωτοκόλλων και εκμεταλλεύονται τους αμυντικούς μηχανισμούς προκειμένου να αναπτύξουν νέες επιθέσεις. [58] [59] [60] [61] [62] [63] [64] [65] [66] [67]

2.2.14 **Πειρατεία** DNS (DNS Hijacking)

Όπως είναι γνωστό, το Domain Name System (DNS), είναι κυρίως υπεύθυνο για την μετάφραση ενός φιλικού προς τον χρήστη ονόματος ιστοσελίδας (domain name), όπως το "Google.com" στην αντίστοιχη διεύθυνση IP του "74.125.235.46". Ο DNS εξυπηρετητής (server) ανήκει και διατηρείται από τον πάροχο υπηρεσιών διαδικτύου (Internet Service Provider (ISP)) του χρήστη και από διάφορες άλλες ιδιωτικές επιχειρήσεις / οργανισμούς. Από προεπιλογή, ο υπολογιστής του χρήστη είναι ρυθμισμένος να χρησιμοποιεί τον DNS εξυπηρετητή του ISP. Σε ορισμένες περιπτώσεις, ο υπολογιστής ενός χρήστη είναι πιθανό να χρησιμοποιεί τις υπηρεσίες DNS ενός άλλου πάροχου, όπως π.χ. της Google. Σε αυτή την περίπτωση, όλα είναι υπό έλεγχο και δεν υπάρχει κανένας λόγος ανησυχίας. Ωστόσο, υπάρχουν και περιπτώσεις όπου ένας hacker ή κάποιο κακόβουλο λογισμικό κερδίζει μη εξουσιοδοτημένη πρόσβαση στον υπολογιστή του χρήστη και αλλάζει τις DNS ρυθμίσεις του.

Το DNS Hijacking (πολλές φορές αναφέρεται και ως DNS Redirection) είναι ένα είδος κακόβουλης επίθεσης που παρακάμπτει τις TCP/IP ρυθμίσεις στον υπολογιστή ενός χρήστη προκειμένου να τον ανακατευθύνει σε έναν κακόβουλο εξυπηρετητή, ακυρώνοντας τις προεπιλεγμένες DNS ρυθμίσεις του υπολογιστή του. Με απλά λόγια, όταν ο χρήστης πληκτρολογεί το όνομα της ιστοσελίδας (domain name) που θέλει να επισκεφτεί δεν μεταφράζεται σωστά η διεύθυνση IP με αποτέλεσμα να μεταφερθεί σε μια κακόβουλη ιστοσελίδα που ανήκει σε έναν hacker.

Οι κίνδυνοι από αυτή την επίθεση μπορεί να ποικίλουν και εξαρτώνται από τα κίνητρα του επιτιθέμενου. Πολλές φορές ο επιτιθέμενος το κάνει για να βλάψει την φήμη μιας ιστοσελίδας, για να υλοποιήσει επιθέσεις όπως Pharming και Phishing.

Το **Pharming** είναι ένα είδος επίθεσης όπου ο χρήστης αντί να μεταφερθεί στην ιστοσελίδα που θέλει, ανακατευθύνεται σε μια άλλη που συνήθως είναι γεμάτη με αναδυόμενα παράθυρα (pop-ups) και διαφημίσεις. Από τις διαφημίσεις αυτές οι hackers δημιουργού έσοδα.

Το **Ψάρεμα (Phishing)** είναι ένα είδος επίθεσης όπου ο χρήστης αντί να μεταφερθεί στην ιστοσελίδα που θέλει, ανακατευθύνεται σε μια άλλη ιστοσελίδα, η οποία ουσιαστικά αποτελεί ένα πιστό αντίγραφο της ιστοσελίδας που θέλει να επισκεφθεί αρχικά ο χρήστης. Ένα κλασικό παράδειγμα, είναι η ανακατεύθυνση ενός θύματος σε μια κακόβουλη ιστοσελίδα που αποτελεί πιστό αντίγραφο της ιστοσελίδας της τράπεζας με την οποία συνεργάζεται. Το θύμα, χωρίς να το καταλάβει, εισάγει τα συνθηματικά του (username & password) και αντί να συνδεθεί ανακατευθύνεται στην πραγματική ιστοσελίδα έχοντας την εντύπωση ότι πληκτρολόγησε κάτι λάθος. Παράλληλα, τα συνθηματικά του θύματος αποστέλλονται στον ιδιοκτήτη της κακόβουλης σελίδας.

Εικόνα 2.7 - Phishing Attack [153]

Για να προστατευτεί κάποιος χρήστης από αυτές τις επιθέσεις θα πρέπει να αποφεύγει μη αξιόπιστες ιστοσελίδες που παρέχουν δωρεάν λογισμικό. Πολλές φόρες, ένας χρήστης μπορεί να κάνει εγκατάσταση ενός προγράμματος και χωρίς να το καταλάβει να εγκατασταθεί ταυτόχρονα κάποιο κακόβουλο λογισμικό, γνωστό και ως δούρειος ίππος (Trojan Horse), το οποίο μπορεί να επιτρέψει σε έναν hacker την εξουσιοδοτημένη πρόσβαση στον υπολογιστή του θύματος με σκοπό να αλλάξει τις DNS ρυθμίσεις του. Το DNSChanger Trojan είναι ένα χαρακτηριστικό παράδειγμα κακόβουλου λογισμικού που άλλαξε τις DNS ρυθμίσεις σε 4 εκατομμύρια υπολογιστές.

Είναι εξίσου σημαντικό για την ασφάλεια ενός χρήστη να αλλάξει τον αρχικό κωδικό του δρομολογητή (router) του και να χρησιμοποιεί κάποιο πρόγραμμα προστασίας (antivirus) το οποίο θα πρέπει να είναι συνεχώς ενημερωμένο. Θα πρέπει να είναι ιδιαίτερα προσεκτικός απέναντι στα email που στέλνουν άγνωστοι χρήστες. Σε καμία περίπτωση δεν θα πρέπει να αποθηκεύει αρχεία και να επισκέπτεται προτεινόμενους συνδέσμους από email που δεν γνωρίζει. Επιπροσθέτως, δεν θα πρέπει να δίνει τα προσωπικά του στοιχεία σε μη αξιόπιστες ιστοσελίδες και αναδυόμενα παράθυρα. [68] [69] [99] [100]

2.2.15 **Πλαστογράφηση** DNS (DNS Spoofing)

Η πλαστογράφηση DNS είναι ένας τύπος επίθεσης κατά την οποία κακόβουλα δεδομένα εισάγονται στον DNS εξυπηρετητή (server). Η επίθεση αυτή είναι γνωστή ως cache Poisoning / DNS cache Poisoning και έχει τα ίδια αποτελέσματα με μια επίθεση **Πειρατείας DNS (DNS Hijacking)**. Ωστόσο, η εν λόγω επίθεση διαφέρει από την επίθεση Πειρατείας DNS (DNS Hijacking) επειδή έχει σαν στόχο τον DNS εξυπηρετητή και όχι τον υπολογιστή του θύματος. Πιο εξελιγμένες χρήσεις αυτής της επίθεσης περιλαμβάνουν επιθέσεις **άρνησης υπηρεσίας (Denial of Service Attack)** και επιθέσεις **Man-in-the-middle (MITM)**.

Η επιτυχία αυτής της επίθεσης στηρίζεται στην εκμετάλλευση των τρωτών σημείων που βρίσκονται στο λογισμικό του DNS εξυπηρετητή. Μόλις ο επιτιθέμενος στείλει μια πλαστή DNS απάντηση (response), τα κακόβουλα δεδομένα αποθηκεύονται προσωρινά στην μνήμη cache του DNS εξυπηρετητή. Σε αυτό ακριβώς το σημείο, είναι που η μνήμη cache θεωρείται "δηλητηριασμένη". Κατά συνέπεια, οι χρήστες που θα θελήσουν να επισκεφτούν το όνομα χώρου (domain name) που έχει δεχτεί επίθεση, θα ανακατευθυνθούν σε κάποιο άλλο όνομα χώρου. Αυτό θα συμβεί επειδή ο επιτιθέμενος θα έχει αντικαταστήσει την αρχική διεύθυνση IP με μία άλλη, η οποία θα παραπέμπει τους χρήστες στην κακόβουλη ιστοσελίδα. Η επίθεση θα σταματήσει μόνο όταν γίνει εκκαθάριση της μνήμης cache.

Δυστυχώς, για την προστασία από αυτή την επίθεση δεν μπορεί να κάνει κάτι ένας προγραμματιστής διαδικτυακών εφαρμογών. Ωστόσο, οι κάτοχοι DNS εξυπηρετητών μπορούν να συμβάλουν στην αποτροπή αυτής της επίθεσης με το να βασίζονται όσο το δυνατόν λιγότερο σε άλλους DNS εξυπηρετητές. Πέρα από τον περιορισμό των σχέσεων εμπιστοσύνης μεταξύ των DNS εξυπηρετητών πρέπει:

- ✓ Να χρησιμοποιούν αναβαθμισμένο λογισμικό.
- ✓ Να περιορίζουν αναδρομικά ερωτήματα (queries).
- ✓ Να αποθηκεύουν μόνο τα δεδομένα που σχετίζονται με το αιτούμενο όνομα (domain)
- ✓ Να επιτρέπουν τις απαντήσεις σε ερωτήματα (queries) που αφορούν μόνο πληροφορίες σχετικά με το αιτούμενο όνομα (domain)
- ✓ Να απενεργοποιούνται οι επιπρόσθετες υπηρεσίες που δεν είναι απαραίτητες.

Επιπροσθέτως, υπάρχουν διαθέσιμα εργαλεία για την πρόβλεψη αυτής της επίθεσης. Το πιο δημοφιλές είναι το DNSSEC (Domain Name System Security Extension). Το DNSSEC έχει αναπτυχθεί από τον οργανισμό IETF (Internet Engineering Task Force). [70] [71] [72]

2.2.16 **Λήψη αποτυπωμάτων (**Fingerprinting)

Η λήψη αποτυπωμάτων είναι μια γνωστή μέθοδος που χρησιμοποιούν οι εισβολείς προκειμένου να συλλέξουν πληροφορίες για το σύστημα που θέλουν να βλάψουν. Το πλαίσιο της επίθεσης διαμορφώνεται με βάση αυτές τις πληροφορίες και για να γίνει αυτό απαιτούνται τα εξής:

- ✓ Ο προσδιορισμός της αρχιτεκτονικής / τοπολογίας της εφαρμογής
- ✓ Ο προσδιορισμός της έκδοσης του εξυπηρετητή της εφαρμογής
- ✓ Ο προσδιορισμός του λογισμικού της εφαρμογής του εξυπηρετητή
- ✓ Ο προσδιορισμός της Βάσης Δεδομένων
- ✓ Ο προσδιορισμός της τεχνολογίας των υπηρεσιών που χρησιμοποιούνται

Η μέθοδος της λήψης αποτυπωμάτων πολλαπλών επιπέδων είναι παρόμοια με την μέθοδο της λήψης αποτυπωμάτων TCP/IP, (π.χ. με χρήση του σαρωτή nmap) με την διαφορά ότι εστιάζει στο επίπεδο της εφαρμογής (Application Layer 7) του μοντέλου OSI αντί να εστιάσει στο επίπεδο μεταφοράς (Transport Layer 4). Απαιτεί τη συλλογή των πληροφοριών που προαναφέρθηκαν καθώς και την αναγνώριση της αρχιτεκτονικής του δικτύου (αν είναι εφικτό).

Επομένως, όσο λιγότερα στοιχεία γνωρίζει κάποιος για το σύστημα τόσο μειώνονται οι πιθανότητες για τον εντοπισμό αδυναμιών και την υλοποίηση επιθέσεων. [7]

2.2.17 **Συμβολοσειρά μορφοποίησης (**Format String)

Οι επιθέσεις που στοχεύουν στη συμβολοσειρά μορφοποίησης μπορούν να αλλάζουν την ροή μιας διαδικτυακής εφαρμογής. Οι επιθέσεις αυτές χρησιμοποιούν τα χαρακτηριστικά των βιβλιοθηκών των συμβολοσειρών μορφοποίησης για να αποκτήσουν πρόσβαση σε άλλους χώρους μέσα στη μνήμη. Ο εισβολέας εκμεταλλεύεται την ευπάθεια ακατάλληλου χειρισμού εισόδου και εισάγει δεδομένα τα οποία χρησιμοποιούνται απευθείας ως συμβολοσειρές μορφοποίησης σε ορισμένες συναρτήσεις C/C++ (π.χ. fprintf, printf, sprintf, setproctitle, syslog, κλπ.).

Αν ο εισβολέας καταφέρει να εισάγει μια συμβολοσειρά που περιλαμβάνει χαρακτήρες μετατροπής (π.χ. για την printf "% f", "%" p, "% n", κλπ.) ως τιμή παραμέτρου σε μια διαδικτυακή εφαρμογή, και αυτή η παράμετρος χρησιμοποιηθεί ως συμβολοσειρά μορφοποίησης, τότε μπορούν να συμβούν τα παρακάτω:

- ✓ Εκτέλεση αυθαίρετου κώδικα στον εξυπηρετητή.
- ✓ Ανάγνωση των τιμών στη στοίβα.
- ✓ Πρόκληση σφαλμάτων κατάτμησης και διακοπή εκτέλεσης του λογισμικού.

Η υπερχείλιση ενδιάμεσης μνήμης (Buffer Overflow) και η υπερχείλιση ακεραίων (Integer Overflow) είναι δύο επιθέσεις που συνδέονται με την επίθεση αυτή. Και οι τρείς επιθέσεις βασίζονται στην ικανότητά τους να διαχειρίζονται τη μνήμη ή το διερμηνευτή του συστήματος με τέτοιο τρόπο που να συμβάλλουν στη μη ομαλή λειτουργία της εφαρμογής. [7]

2.2.18 Google Hacking

Ο όρος Google Hacking χρησιμοποιείται όταν ένας εισβολέας προσπαθεί να εντοπίσει εκμεταλλεύσιμους στόχους και ευαίσθητα δεδομένα κάνοντας χρήση μιας μηχανής αναζήτησης. Η Google Hacking Database (GHDB) είναι μια Βάση Δεδομένων που περιλαμβάνει ερωτήματα (queries) τα οποία προσδιορίζουν ευαίσθητα δεδομένα. Παρόλο που η Google μπλοκάρει ορισμένα από τα πιο γνωστά Google Hacking ερωτήματα (queries), δεν σημαίνει ότι ένας εισβολέας δεν μπορεί να εντοπίσει την εφαρμογή και να ξεκινήσει να υποβάλει τα αντίστοιχα ερωτήματα.

Η GHDB περιλαμβάνει πληροφορίες σχετικά:

- ✓ Με τα τρωτά σημεία του εξυπηρετητή
- ✓ Με τα μηνύματα σφαλμάτων, τα οποία περιλαμβάνουν σημαντικές πληροφορίες
- ✓ Με τα αρχεία που περιέχουν κωδικούς πρόσβασης
- ✓ Με ευαίσθητες διαδρομές (directories)
- ✓ Με σελίδες που περιέχουν πύλες σύνδεσης
- ✓ Με σελίδες που περιέχουν δεδομένα για το δίκτυο ή για ευπάθειες που αφορούν το τείχος προστασίας (firewall)

Ο ευκολότερος τρόπος για να ελέγξει ένας προγραμματιστής αν μια διαδικτυακή εφαρμογή είναι ευάλωτη σε αυτή την επίθεση, είναι να χρησιμοποιήσει έναν σαρωτή ευπαθειών ο οποίος θα μπορεί να εκτελέσει και Google Hacking ερωτήματα (queries). Ένας τέτοιος σαρωτής είναι ο Acunetix.

Για την πρόληψη από αυτή την επίθεση θα πρέπει να επιβεβαιωθεί ότι όλες οι σελίδες ελέγχθηκαν από Google Hacking ερωτήματα (queries). Δεδομένου ότι υπάρχουν σελίδες που παρέχουν πληροφορίες, οι οποίες δε θα έπρεπε να είναι διαθέσιμες, ο προγραμματιστής οφείλει να τις αφαιρέσει από την ιστοσελίδα. Σε περίπτωση που αυτές οι σελίδες είναι απαραίτητες να υπάρχουν, θα πρέπει να μην εμφανίζονται στα αποτελέσματα των μηχανών αναζήτησης και να διατυπώνεται το περιεχόμενό τους με τέτοιο τρόπο που να μην είναι εύκολο να εντοπιστεί από τα Google Hacking ερωτήματα (queries). [73]

2.2.19 **Έγχυση επικεφαλίδας** HTTP (HTTP Header Injection)

Η έγχυση σε επικεφαλίδα HTTP είναι μια γενική κατηγορία επιθέσεων που εκμεταλλεύονται την ευπάθεια του **ακατάλληλου χειρισμού εισόδου (Improper Input Handling)** των δεδομένων του χρήστη, τα οποία χρησιμοποιούνται για την αναπαραγωγή δυναμικών επικεφαλίδων HTTP.

Μια επίθεση έγχυσης επικεφαλίδας μπορεί να επιτρέψει επιθέσεις:

- Διάσπασης απαντήσεων HTTP (γνωστές και ως CRLF)
- Κακόβουλου ορισμού αναγνωριστικών συνόδου (Session fixation)
- Cross-Site Scripting (XSS) καθώς και
- επιθέσεις ανακατεύθυνσης

Ευτυχώς, ευπάθειες λόγω επιθέσεων έγχυσης επικεφαλίδας HTTP δεν είναι πλέον εφικτές. Το γεγονός αυτό οφείλεται στην δυνατότητα εμποδισμού πολλαπλών αιτήσεων επικεφαλίδας. [74]

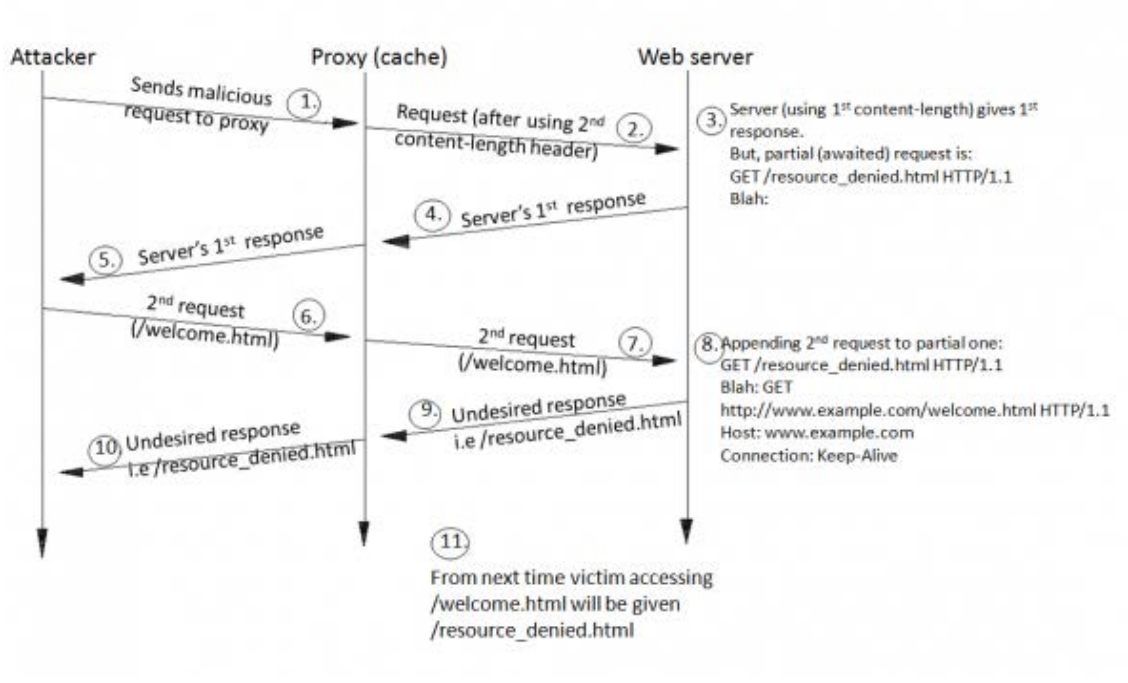
2.2.20 **Λαθραία αιτήματα** HTTP (HTTP Request Smuggling)

Τα λαθραία αιτήματα HTTP είναι μία μέθοδος επίθεσης που βασίζεται στην ελλιπή ανάλυση κακόβουλων HTTP αιτημάτων που υποβάλλονται σε ένα σύστημα το οποίο απαρτίζεται από πολλαπλές οντότητες. Ένα τέτοιο σύστημα μπορεί να απαρτίζεται από κάποιο ενδιάμεσο εξυπηρετητή (proxy) και από τον εξυπηρετητή (server). Τα αιτήματα αυτά δεν υπακούν στους κανόνες του πρωτόκολλου HTTP και αυτό έχει σαν αποτέλεσμα να αναλύονται με διαφορετικό τρόπο από τον ενδιάμεσο εξυπηρετητή (proxy) και από τον εξυπηρετητή (server). Με αυτό τον τρόπο ο εισβολέας μπορεί να εξαπατήσει το σύστημα χωρίς να γίνει αντιληπτός.

Η επίθεση αυτή μπορεί να παρακάμψει το τείχος προστασίας (firewall) καθώς επίσης και να ευνοήσει επιθέσεις όπως:

- ✓ Πειρατεία συνόδου (Session Hijacking)
- ✓ Cross-Site Scripting (XSS)
- ✓ Cache Poisoning

Υπάρχουν αρκετά σενάρια που μπορεί να εφαρμόσει ένας εισβολέας για την υλοποίηση αυτής της επίθεσης. Σύμφωνα με ένα απλό σενάριο, ο εισβολέας εκμεταλλεύεται τον ενδιάμεσο εξυπηρετητή (proxy) έτσι ώστε να αντιστοιχίσει το περιεχόμενο μιας διεύθυνσης URL σε μια άλλη. Έτσι, όταν π.χ. ο χρήστης πληκτρολογεί το URL A, εμφανίζεται το περιεχόμενο που θα εμφανίζονταν αν πληκτρολογούσε το URL B, ενώ αν πληκτρολογούσε το URL B ο ενδιάμεσος εξυπηρετητής (proxy) θα εμφάνιζε το περιεχόμενο που αντιστοιχεί στο URL A. [75] [76]



Εικόνα 2.8 - HTTP Request Smuggling [154]

Για την προστασία από αυτή την επίθεση θα πρέπει:

- ✓ Να εγκατασταθεί ένα τείχος προστασίας (firewall), που να είναι κατάλληλο για την αποτροπή της εν λόγω επίθεσης. Άξιο αναφοράς είναι το γεγονός ότι υπάρχουν ακόμα μερικά firewalls που είναι ευάλωτα στη συγκεκριμένη επίθεση.
- ✓ Να εφαρμόζονται αυστηρές τεχνικές όσον αφορά τη διαχείριση των συνόδων (sessions).
- ✓ Η επικοινωνία μεταξύ χρήστη και εξυπηρετητή να γίνεται μόνο μέσα από τη χρήση ενός SSL πρωτόκολλου.
- ✓ Να απενεργοποιηθεί η κοινή χρήση σύνδεσης TCP μεταξύ των ενδιάμεσων συσκευών. Η κοινόχρηστη σύνδεση TCP βελτιώνει την απόδοση, αλλά επιτρέπει στους εισβολείς να περάσουν λαθραία αιτήματα HTTP.
- ✓ Να χρησιμοποιείται ένας εξυπηρετητής που χρησιμοποιεί μια αυστηρότερη διαδικασία ανάλυσης HTTP, όπως π.χ. ο Apache

2.2.21 **Διάσπαση αιτήσεων** HTTP (HTTP Request Splitting)

Η διάσπαση αιτήσεων HTTP είναι μια επίθεση που αναγκάζει τον περιηγητή του χρήστη να στέλνει αυθαίρετες HTTP αιτήσεις, προκαλώντας επιθέσεις **Cross-site Scripting (XSS)** καθώς και προβλήματα στη κρυφή μνήμη (cache) του περιηγητή. Για να γίνει αυτό, θα πρέπει ο χρήστης να επισκεφτεί την κακόβουλη ιστοσελίδα του επιτιθέμενου προκειμένου να κάνει τον περιηγητή του χρήστη να στείλει δύο HTTP αιτήσεις αντί για μία. Η αποστολή αυτών των HTTP αιτήσεων οφείλεται στην εκμετάλλευση των παρακάτω μηχανισμών του περιηγητή:

- ✓ XmlHttpRequest object
- ✓ HTTP digest authentication

Για να πραγματοποιηθεί η εν λόγω επίθεση θα πρέπει ο περιηγητής του χρήστη να χρησιμοποιεί έναν αντιπρόσωπο προώθησης HTTP (HTTP proxy), διαφορετικά θα πρέπει χρήστης και επιτιθέμενος να χρησιμοποιούν την ίδια διεύθυνση IP. Αν τίποτα από τα δύο δεν συμβαίνει η εν λόγω επίθεση δεν μπορεί να πραγματοποιηθεί.

Αν και η διάσπαση αιτήσεων HTTP είναι μια πολύ σπάνια επίθεση, οι ακόλουθες συστάσεις θα πρέπει να ληφθούν σοβαρά: [76]

- ➢ Είναι καλό για τους ιδιοκτήτες ιστοσελίδων να χρησιμοποιούν SSL για προστασία.
- ➢ Θα πρέπει να λαμβάνονται τα κατάλληλα μέτρα για την πρόληψη των **Cross-site Scripting (XSS)** επιθέσεων.
- ➢ Θα πρέπει να αποκλείονται αιτήσεις HTTP/1.0 προς τον εξυπηρετητή. Κάτι τέτοιο μπορεί να λειτουργήσει θετικά από πλευράς ασφάλειας αλλά αρνητικά προς τις μηχανές αναζήτησης, οι οποίες χρησιμοποιούν HTTP/1.0.
- ➢ Θα πρέπει να ακολουθούνται οι συμβουλές προστασίας που προτείνονται για τις ακόλουθες επιθέσεις:
  - ✓ **Λαθραίες απαντήσεις HTTP** (HTTP Response Smuggling)
  - ✓ **Διάσπαση απαντήσεων HTTP** (HTTP Response Splitting)

2.2.22 **Λαθραίες απαντήσεις** HTTP (HTTP Response Smuggling)

Πρόκειται για μια τεχνική επίθεσης κατά την οποία στέλνονται δύο απαντήσεις HTTP από τον εξυπηρετητή στο χρήστη μέσω μιας ενδιάμεσης συσκευής HTTP, η οποία αναμένει (ή επιτρέπει) μια μόνο απάντηση από τον εξυπηρετητή.

Οι επιθέσεις που βασίζονται στις λαθραίες απαντήσεις HTTP χρησιμοποιούνται για να παρακάμψουν τα μέτρα ασφαλείας που χρησιμοποιούνται για την προστασία από επιθέσεις

διάσπασης απαντήσεων HTTP (HTTP Response Splitting ή HRS). Σε αυτή την περίπτωση, η ενδιάμεση συσκευή είναι ο μηχανισμός προστασίας κατά των HRS επιθέσεων.

Μια άλλη χρήση αυτής της επίθεσης είναι να πλαστογραφηθούν οι απαντήσεις που λαμβάνονται από τον περιηγητή. Σε αυτή την περίπτωση, μια κακόβουλη ιστοσελίδα στέλνει στον περιηγητή μια σελίδα, που την ερμηνεύει ως μια σελίδα που ανήκει σε διαφορετική ιστοσελίδα. Αυτό μπορεί να συμβεί όταν ο περιηγητής χρησιμοποιεί έναν ενδιάμεσο (proxy) εξυπηρετητή για να προσπελάσει και τις δύο ιστοσελίδες.

Για να προστατευτεί ένας χρήστης από αυτή την επίθεση θα πρέπει να κάνει τις απαραίτητες ρυθμίσεις στο περιηγητή του προκειμένου να μην επιτρέπονται οι μη έγκυρες ή διφορούμενες απαντήσεις. Θα ήταν ιδανικό αν μετατρέπονταν οι απαντήσεις σε μηνύματα σφάλματος (ίσως με κωδικό κατάστασης 5xx) και να τερματίζονταν η TCP σύνδεση. Από την πλευρά του προγραμματιστή, θα πρέπει να κωδικοποιούνται οι πληροφορίες που προέρχονται από την εισαγωγή δεδομένων του χρήστη, και χρησιμοποιούνται σε επικεφαλίδες. [76] [77]

2.2.23 **Διάσπαση απαντήσεων** HTTP (HTTP Response Splitting)

Για να επιτευχθεί αυτή η επίθεση θα πρέπει να υπάρχει κενό ασφαλείας στον εξυπηρετητή (server) που να επιτρέπει τη διάσπαση απαντήσεων HTTP. Ο στόχος της επίθεσης μπορεί να είναι είτε ένας ενδιάμεσος εξυπηρετητής (cache forward/ reverse proxy) είτε μια εφαρμογή πλοήγησης (πιθανότατα με κρυφή μνήμη).

Πρόκειται για μια τεχνική που επιτρέπει σε έναν εισβολέα την αποστολή μιας απλής HTTP αίτησης (request) που αναγκάζει τον εξυπηρετητή να δημιουργήσει μια ροή εξόδου (output stream), η οποία στη συνέχεια ερμηνεύεται από το σύστημα-στόχο σαν δύο HTTP απαντήσεις αντί για μια. Η πρώτη απάντηση μπορεί να ελέγχεται εν μέρει από τον εισβολέα, αλλά δεν έχει σημασία. Αυτό που έχει σημασία είναι ότι ο εισβολέας είναι σε θέση να ελέγξει τη δομή της δεύτερης απάντησης από την γραμμή κατάστασης HTTP μέχρι και το τελευταίο byte αυτής της απάντησης. Από τη στιγμή που μπορεί να συμβεί κάτι τέτοιο, ο εισβολέας πραγματοποιεί αυτή την επίθεση στέλνοντας δύο αιτήματα μέσω του στόχου.

Το πρώτο αίτημα προκαλεί δύο απαντήσεις από τον εξυπηρετητή ιστού και το δεύτερο αίτημα μπορεί να είναι τυπικά ένας «αθώος» πόρος για τον εξυπηρετητή. Παρόλα αυτά, το δεύτερο αίτημα μπορεί να ταιριάζεται από το στόχο με τη δεύτερη HTTP απάντηση, η οποία ελέγχεται από τον εισβολέα. Με αυτό τον τρόπο ο εισβολέας εξαπατάει το στόχο και τον οδηγεί να πιστέψει ότι ένας συγκεκριμένος πόρος στον εξυπηρετητή (ο οποίος ζητήθηκε με το δεύτερο αίτημα) αντιστοιχεί σε κάποια δεδομένα τα οποία έχουν δημιουργηθεί από τον εισβολέα μέσα από τον εξυπηρετητή (το δεύτερο τμήμα της πρώτης απάντησης).

Μια επίθεση διάσπασης απαντήσεων HTTP μπορεί να χρησιμοποιηθεί για την υλοποίηση επιθέσεων όπως:

- **Cross-Site Scripting (XSS)**
- **Cross-User Defacement**
- **Cache Poisoning** κ.α.

Ευτυχώς, με την πρόληψη της ευπάθειας του **ακατάλληλου χειρισμού εισόδου (Improper Input Handling)** μπορούν να αντιμετωπιστούν οι επιθέσεις διάσπασης απαντήσεων HTTP. Είναι σημαντικό να ελέγχονται και να κωδικοποιούνται σωστά τα δεδομένα που προέρχονται από την εισαγωγή δεδομένων του χρήστη, τα οποία πρόκειται να χρησιμοποιηθούν σε επικεφαλίδες HTTP. Είναι εξίσου σημαντικό να αφαιρεθούν χαρακτήρες CR (Carriage Return) και LF (Line Feed), οι οποίοι ευνοούν αυτή την επίθεση. Η επίθεση αυτή είναι γνωστή ως έγχυση CRLF (Carriage Return Line Feed Injection).

Τέλος, η ασφάλεια της εφαρμογής δε θα πρέπει να στηρίζεται στη χρήση SSL, αφού δεν μπορεί να προστατεύσει την εφαρμογή από αυτή την επίθεση.

Άξιο αναφοράς είναι το γεγονός ότι παλαιότερα, η υπηρεσία Google Adwords βρέθηκε αντιμέτωπη με επιθέσεις έγχυσης CRLF. [78] [79] [80] [76] [81]

2.2.24 **Υπερχείλιση ακεραίων** (Integer Overflows)

Μια επίθεση υπερχείλισης ακεραίων πραγματοποιείται όταν το αποτέλεσμα που προκύπτει από μια αριθμητική λειτουργία, όπως από μια πρόσθεση ή έναν πολλαπλασιασμό, υπερβαίνει τη μέγιστη τιμή που μπορεί να δοθεί και να αποθηκευτεί σε μια μεταβλητή. Η μέγιστη τιμή εξαρτάται από τον τύπο του ακεραίου που χρησιμοποιείται και όταν ξεπερνιέται το μέγιστο όριο, η τιμή που θα αποθηκευτεί θα είναι η ελάχιστη τιμή που μπορεί να πάρει ο χρησιμοποιούμενος τύπος ακεραίου και όχι η πραγματική τιμή που προκύπτει από το αποτέλεσμα της αριθμητικής λειτουργίας.

Για παράδειγμα ένας ακέραιος αριθμός 8bit στην αρχιτεκτονική των περισσότερων υπολογιστών έχει μια μέγιστη τιμή 127 και μια ελάχιστη -128. Στην περίπτωση που ο προγραμματιστής έχει αποθηκεύσει την τιμή 127 σε μια τέτοια μεταβλητή και θελήσει αργότερα να προσθέσει σε αυτή την τιμή 1, τότε το αποτέλεσμα που θα προκύψει δεν θα είναι 128 αλλά -128.

Για την προστασία από αυτή την επίθεση θα πρέπει να γίνονται οι απαραίτητοι έλεγχοι των δεδομένων που εισάγονται στην εφαρμογή. Επιπλέον, θα πρέπει να γίνονται οι απαραίτητοι έλεγχοι για σφάλματα (bugs) κατά την υλοποίηση της εφαρμογής για να εξασφαλιστεί η σωστή χρήση των μεταβλητών. [7]

2.2.25 Έγχυση στον LDAP (LDAP Injection)

Το Lightweight Directory Access Protocol (LDAP) είναι ένα ευρέως χρησιμοποιούμενο ανοιχτό πρωτόκολλο τόσο για την αναζήτηση όσο και για το χειρισμό των υπηρεσιών καταλόγου X.500. Το πρωτόκολλο LDAP εκτελείται πάνω από τα πρωτόκολλα του επιπέδου μεταφοράς, όπως το TCP. Οι εφαρμογές διαδικτύου μπορούν να χρησιμοποιήσουν τα δεδομένα που εισάγει ο χρήστης (input) προκειμένου να παράγουν LDAP εντολές για την απάντηση σε διάφορα αιτήματα μέσα από δυναμικές σελίδες.

Η LDAP Injection (LDAPI) είναι μια τεχνική που εκμεταλλεύεται την ευπάθεια του ακατάλληλου χειρισμού εισόδου του χρήστη και έχει σαν αποτέλεσμα την εκτέλεση αυθαίρετων εντολών στην εφαρμογή. Όταν ο εισβολέας πραγματοποιήσει μια τέτοια ενέργεια, η εντολή που θα εκτελεστεί θα έχει τα ίδια δικαιώματα (permissions) με αυτά που έχει το στοιχείο που εκτέλεσε την εντολή (π.χ. ο εξυπηρετητής της Βάσης Δεδομένων, ο εξυπηρετητής της εφαρμογής κλπ.). [82]

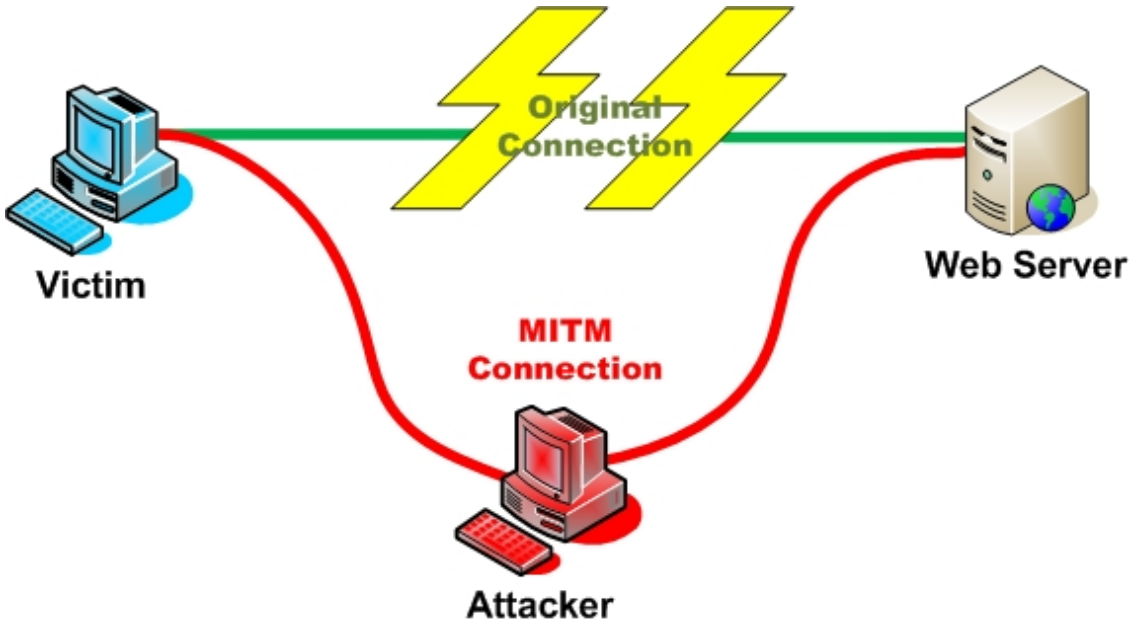
2.2.26 Έγχυση στο σύστημα ηλεκτρονικής αλληλογραφίας (Mail Command Injection)

Η έγχυση στο σύστημα ηλεκτρονικής αλληλογραφίας είναι μια τεχνική επίθεσης που εφαρμόζεται ενάντια σε εξυπηρετητές ηλεκτρονικού ταχυδρομείου και εφαρμογές webmail που δεν προστατεύονται από την ευπάθεια του ακατάλληλου χειρισμού εισόδου. Ο εισβολέας επιδιώκει την αναπαραγωγή λανθασμένων IMAP/SMTP εντολών για να αποκτήσει πρόσβαση σε κάποιον εξυπηρετητή ηλεκτρονικού ταχυδρομείου.

Η έγχυση στο σύστημα ηλεκτρονικής αλληλογραφίας είναι γνωστή και ως IMAP/ SMTP έγχυση (IMAP/ SMTP Command Injection). Ανάλογα με τον τύπο του εξυπηρετητή (IMAP/SMTP server), συναντάμε δύο τύπους έγχυσης, την **έγχυση IMAP** και την **έγχυση SMTP**. Στην περίπτωση μιας έγχυσης SMTP, ο εισβολέας θα πρέπει να διαθέτει έναν έγκυρο λογαριασμό webmail. Αυτό σημαίνει ότι μια έγχυση SMTP μπορεί να πραγματοποιηθεί μόνο από έναν πιστοποιημένο χρήστη. Από την άλλη πλευρά, μια έγχυση IMAP μπορεί να επιτευχθεί από έναν μη πιστοποιημένο χρήστη, αλλά θα έχει πρόσβαση περιορισμένες δυνατότητες. [7]

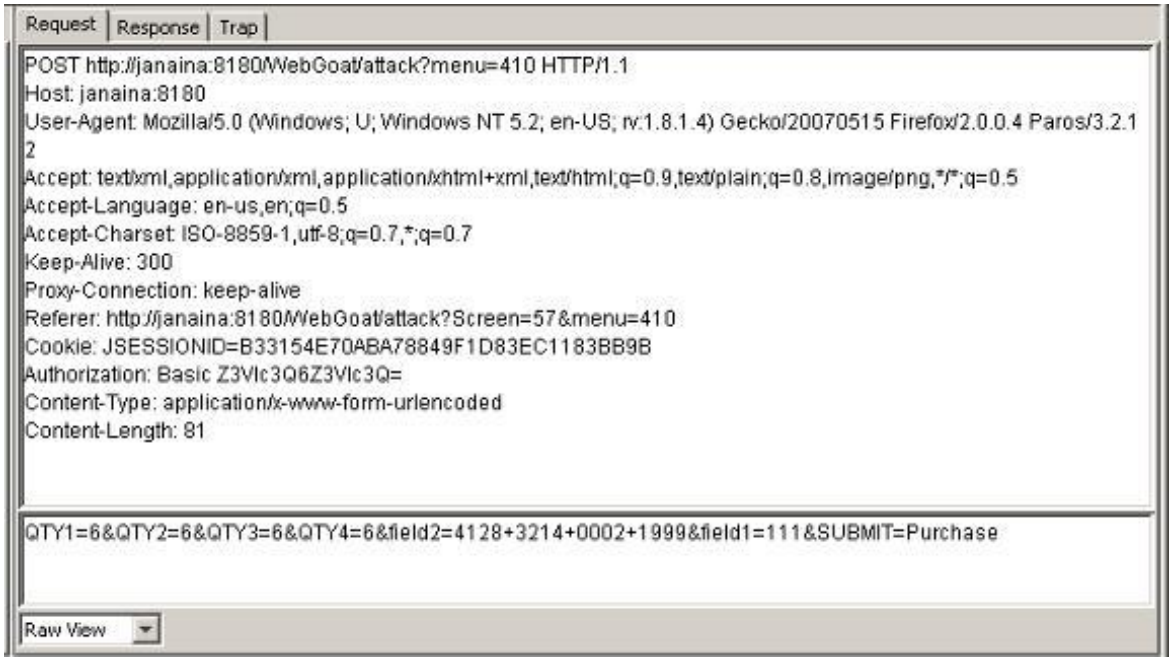
2.2.27 Man-in-the-middle (MITM)

Η επίθεση man-in-the-middle στοχεύει στην TCP σύνδεση μεταξύ του χρήστη (client) και του εξυπηρετητή (server). Σε μια http συναλλαγή, ο επιτιθέμενος προσπαθεί να χωρίσει την αρχική TCP σύνδεση σε δύο νέες συνδέσεις εκ των οποίων, η μία γίνεται μεταξύ του χρήστη και του επιτιθέμενου και η άλλη μεταξύ του επιτιθέμενου και του εξυπηρετητή, όπως φαίνεται στο παρακάτω σχήμα.



Εικόνα 2.9 - Man-in-the-middle Attack [85]

Από την στιγμή που ο επιτιθέμενος μπορεί να παραβιάσει την μεταδιδόμενη κίνηση (traffic) μεταξύ του χρήστη και του εξυπηρετητή, θα είναι σε θέση να διαβάσει, να εισάγει και να τροποποιήσει τα δεδομένα που ανταλλάσουν μεταξύ τους. Η επίθεση MITM είναι πολύ αποτελεσματική λόγω της φύσης του πρωτόκολλου http και της μεταφοράς δεδομένων τα οποία βασίζονται σε κωδικοποίηση ASCII. Έτσι, για παράδειγμα, ο επιτιθέμενος μπορεί να αποκτήσει ένα session cookie με το να διαβάσει την http επικεφαλίδα (header) ή να τροποποιήσει την τιμή κάποιας παραμέτρου που μπορεί να αφορά ένα χρηματικό ποσό καθώς και οποιοδήποτε άλλο ευαίσθητο δεδομένο το οποίο μεταφέρεται μέσα στην εφαρμογή.



Εικόνα 2.10 - Find Session Cookie from HTTP Header [85]

Μια επίθεση MITM θα μπορούσε επίσης να πραγματοποιηθεί μέσω μιας https σύνδεσης, χρησιμοποιώντας την ίδια τεχνική, με την μόνη διαφορά ότι συνιστάται η δημιουργία δυο ανεξάρτητων SSL session, μία για κάθε TCP σύνδεση. Το πρόγραμμα περιήγησης (browser) ορίζει μια SSL σύνδεση με τον εισβολέα, ο οποίος ορίζει ακόμα μια με τον εξυπηρετητή. Συνήθως, ο περιηγητής προειδοποιεί το χρήστη ότι το ψηφιακό πιστοποιητικό (digital certificate) που χρησιμοποιείται δεν είναι έγκυρο, αλλά ο χρήστης πιθανόν να αγνοήσει την προειδοποίηση λόγω του ότι δεν αντιλαμβάνεται την απειλή. Ωστόσο, υπάρχουν περιπτώσεις που δεν εμφανίζεται η αντίστοιχη προειδοποίηση, όπως για παράδειγμα όταν το πιστοποιητικό του εξυπηρετητή έχει παραβιαστεί από τον εισβολέα ή όταν το πιστοποιητικό του εισβολέα είναι υπογεγραμμένο από μια έμπιστη αρχή πιστοποίησης (Certificate Authority) και το CN είναι το ίδιο με αυτό της ιστοσελίδας.

Πολλές φορές η επίθεση MITM υλοποιείται, με χρήση κατάλληλων εργαλείων, είτε κατά το στάδιο της ανάπτυξης μιας διαδικτυακής εφαρμογής είτε μετέπειτα ως τεχνική αξιολόγησης των διαδικτυακών ευπαθειών. [83] [84] [85] [86]

2.2.28 Man-in-the-browser (MITB)

Αυτή η επίθεση είναι μια παραλλαγή της επίθεσης **Man-in-the-middle (MITM)** και επιτυγχάνεται με τη χρήση κακόβουλου λογισμικού που είναι γνωστό ως δούρειος ίππος (Trojan horse). Ο δούρειος ίππος χρησιμοποιείται για να ελέγξει την επικοινωνία μεταξύ του χρήστη και του περιηγητή (browser) και βασίζεται στα κενά ασφάλειας του περιηγητή καθώς και στις δυνατότητες του δούρειου ίππου.

Η επίθεση αυτή μπορεί να εκτελεστεί τόσο σε προσωπικό όσο και σε μαζικό επίπεδο. Για την εγκατάσταση ενός δούρειου ίππου ο εισβολέας πιθανόν να εκμεταλλευτεί:

- Τις ευπάθειες στο δίκτυο
- Τη χρήση του ηλεκτρονικού ταχυδρομείου, με αποστολή μηνυμάτων spam
- Τις ευπάθειες της εφαρμογής
- Επιθέσεις **Phishing,** με χρήση ηλεκτρονικού ταχυδρομείου
- Το διαμοιρασμό συνδέσμων (links) μέσα από ιστοσελίδες κοινωνικής δικτύωσης

Μετά την εγκατάσταση, η ενεργοποίηση γίνεται αυτόματα και ξεκινάει η παρακολούθηση του χρήστη. Παράλληλα με την παρακολούθηση, εκτελούνται και άλλες λειτουργίες που εξαρτώνται από τις ρυθμίσεις που έχει κάνει ο εισβολέας. Αυτές οι λειτουργίες συνήθως ενεργοποιούνται όταν ο χρήστης επισκέπτεται κάποιες συγκεκριμένες ιστοσελίδες (όπως για παράδειγμα ιστοσελίδες τραπεζών).

Πρόκειται για μια εξαιρετικά επικίνδυνη επίθεση λόγω του ότι ένας δούρειος ίππος είναι σχεδιασμένος με τέτοιο τρόπο που δύσκολα μπορεί να ανιχνευτεί. Υπάρχουν αρκετοί, ο «Δίας» (Zeus) είναι ο πιο γνωστός και ο πιο επικίνδυνος.

Είναι σημαντικό για έναν χρήστη να είναι ενημερωμένος για την εν λόγω επίθεση και να μην εγκαθιστά στον υπολογιστή του λογισμικό από μη αξιόπιστους πάροχους. Η χρήση ενός τοίχους (firewall) ή λογισμικού προστασίας (anti-virus) δεν επαρκεί ακόμα και αν ο χρήστης φροντίζει να είναι πάντα ενημερωμένα. Σε περίπτωση που ένας χρήστης θελήσει να επισκεφτεί έναν μη αξιόπιστο ιστότοπο μπορεί να χρησιμοποιήσει μια εικονική μηχανή (virtual machine) η οποία θα πρέπει να είναι κατάλληλα ρυθμισμένη. Είναι εξίσου σημαντική η προστασία του περιηγητή (browser hardening) πράγμα που μπορεί να επιτευχθεί με την χρήση πρόσθετων (extensions) και ειδικά διαμορφωμένου λογισμικού για ασφαλή περιήγηση. Τέλος, θα πρέπει να ελεγχθεί η ασφάλεια του δικτύου ανεξάρτητα από το αν είναι ενσύρματο ή ασύρματο. [87] [88] [89] [90] [91]

2.2.29 **Έγχυση μηδενικών** Bytes (Null Byte Injection)

Η έγχυση μηδενικών bytes είναι μια τεχνική ενεργής εκμετάλλευσης, η οποία χρησιμοποιείται για να παρακάμψει τα λογικά φίλτρα ελέγχου που βρίσκονται στις υποδομές των δικτύων υπολογιστών, προσθέτοντας URL – κωδικοποιημένους χαρακτήρες μηδενικών Byte, (π.χ. %00, ή 0x00 στο δεκαεξαδικό σύστημα αρίθμησης) στα δεδομένα εισόδου των χρηστών. Αυτή η διαδικασία έγχυσης μπορεί να αλλάξει την εσωτερική λογική της εφαρμογής και να επιτρέψει σε κακόβουλους χρήστες να αποκτήσουν πρόσβαση στα αρχεία του συστήματος.

Οι περισσότερες διαδικτυακές εφαρμογές, έχουν αναπτυχθεί χρησιμοποιώντας γλώσσες υψηλού επιπέδου όπως η PHP, ASP, Perl και Java. Παρόλα αυτά, οι εφαρμογές αυτές σε κάποιο βαθμό απαιτούν, επεξεργασία κώδικα υψηλού επιπέδου σε επίπεδο συστήματος και αυτή η επεξεργασία, συνήθως επιτυγχάνεται με τη χρήση συναρτήσεων (functions) C/C++. Η διαφορετική φύση αυτών των ανεξάρτητων τεχνολογιών έχουν οδηγήσει σε μια κατηγορία επίθεσης η οποία λέγεται έγχυση μηδενικών byte (Null Byte Injection) ή δηλητηρίαση μηδενικών byte (Null Byte Poisoning). Στην C/C++ ένα μηδενικό byte αναπαριστά το τερματικό σημείο μιας συμβολοσειράς ή τον χαρακτήρα οριοθέτησης ο οποίος σταματάει την επεξεργασία της συμβολοσειράς άμεσα. Τα bytes τα οποία ακολουθούν τον χαρακτήρα οριοθέτησης αγνοούνται. Εάν η σειρά χάσει το μηδενικό χαρακτήρα, το μήκος της καθίσταται άγνωστο: η συμβολοσειρά εκτείνεται μέχρι ο έλεγχος της μνήμης να συναντήσει το επόμενο μηδενικό byte. Αυτή η διευθέτηση μπορεί να προκαλέσει ασυνήθιστη συμπεριφορά και να παρουσιάσει ευπάθειες μέσα στο σύστημα. Κατά τον ίδιο ή παρόμοιο τρόπο, διάφορες γλώσσες υψηλού επιπέδου διαχειρίζονται τα μηδενικά bytes σαν σύμβολο κράτησης θέσης (placeholder) για το μήκος της συμβολοσειράς, καθώς αυτό δεν έχει συγκεκριμένο νόημα στο περιεχόμενό τους. Εξαιτίας αυτής της διαφοράς στην ερμηνεία τους, τα μηδενικά bytes μπορούν εύκολα να εγχυθούν και να ελέγξουν τη συμπεριφορά της εφαρμογής.

Τα URL περιορίζονται στο να περιέχουν ASCII χαρακτήρες οι οποίοι κυμαίνονται από το 0x20 στο 0x7E (στο δεκαεξαδικό σύστημα αρίθμησης) ή από 32 ως 126 (στο δεκαδικό σύστημα αρίθμησης). Παρόλα αυτά η προαναφερθείσα περιοχή τιμών περιέχει διάφορους χαρακτήρες οι οποίοι δεν επιτρέπονται, επειδή έχουν ξεχωριστή σημασία μέσα στα πλαίσια του HTTP πρωτοκόλλου. Για το λόγο αυτό, το σχήμα κωδικοποίησης URL κατασκευάστηκε για να περιλαμβάνει ειδικούς χαρακτήρες, οι οποίοι χρησιμοποιούν εκτεταμένη αναπαράσταση χαρακτήρων ASCII. Τα μηδενικά bytes κωδικοποιούνται σαν %00 (στο δεκαεξαδικό σύστημα). Ο σκοπός αυτής της επίθεσης ξεκινάει εκεί που οι διαδικτυακές εφαρμογές αλληλεπιδρούν με τις ενεργές ρουτίνες της γλώσσας C και με εξωτερικά APIs (Application Programming Interface) του λειτουργικού συστήματος. Έτσι επιτρέπεται στον εισβολέα να διαχειριστεί διαδικτυακούς πόρους, διαβάζοντας ή γράφοντας αρχεία τα οποία βασίζονται στα δικαιώματα χρήστη της εφαρμογής.

Για την προστασία από αυτή την επίθεση θα πρέπει να ληφθούν μέτρα κατά της ευπάθειας του **ακατάλληλου χειρισμού εισόδου (Improper Input Handling)** των δεδομένων του χρήστη. [92] [93]

2.2.30 **Εκτέλεση εντολών λειτουργικού συστήματος (**OS Commanding)

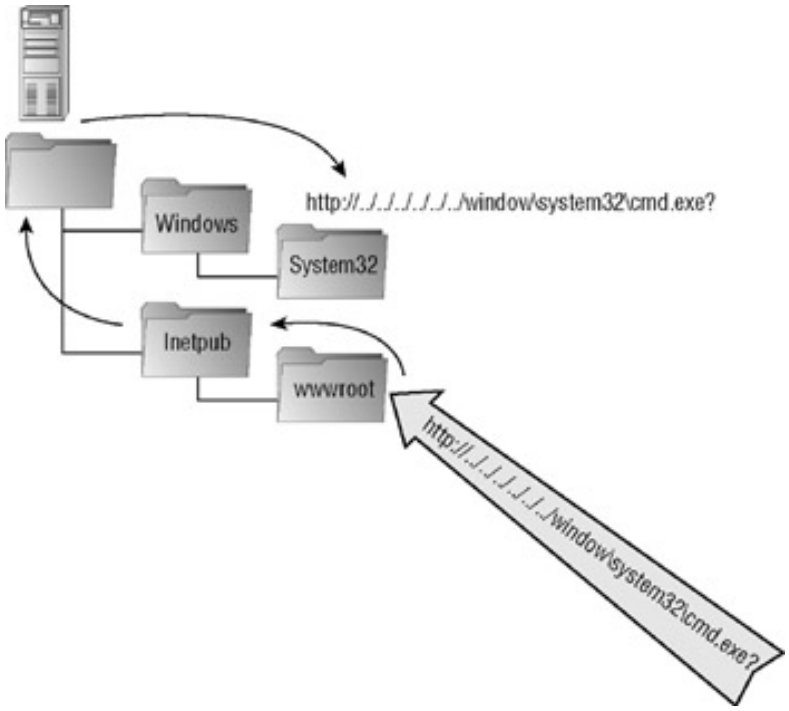
Η OS commanding ή OS Command Injection είναι μια επίθεση που οφείλεται σε **ακατάλληλο χειρισμό εισόδου** και έχει σαν στόχο την μη εξουσιοδοτημένη εκτέλεση εντολών στο λειτουργικό σύστημα. Οι εντολές αυτές συνήθως εκτελούνται με τα ίδια δικαιώματα που εκτελούνται και οι εντολές όπως για παράδειγμα του εξυπηρετητή ιστού, ή του εξυπηρετητή βάσης δεδομένων. Καθώς οι εντολές εκτελούνται με αυξημένα δικαιώματα, ο εισβολέας μπορεί να τις χρησιμοποιήσει για να αποκτήσει πρόσβαση ή να καταστρέψει τμήματα τα οποία υπό άλλες συνθήκες δεν θα μπορούσε. [7]

2.2.31 **Διάσχιση διαδρομής** (Path Traversal)

Μια επίθεση διάσχισης διαδρομής είναι ένας τύπος HTTP εκμετάλλευσης που δίνει τη δυνατότητα στον εισβολέα να αποκτήσει μη εξουσιοδοτημένη πρόσβαση σε απαγορευμένους φακέλους και αρχεία τα οποία βρίσκονται έξω από τον κατάλογο ρίζας (Root Directory). Η εν λόγω επίθεση είναι γνωστή και με τις ακόλουθες ονομασίες:

- ✓ Επίθεση διάσχισης καταλόγου (Directory traversal)
- ✓ Αναρριχόμενη επίθεση (Climbing Attack)
- ✓ Επίθεση υπαναχώρησης (Backtracking Attack)
- ✓ ../ (dot dot slash),

Χρησιμοποιεί το λογισμικό του εξυπηρετητή για να εκμεταλλευτεί ανεπαρκείς μηχανισμούς ασφαλείας και μπορεί να θέσει σε κίνδυνο τόσο την εφαρμογή όσο και τον εξυπηρετητή.



Εικόνα 2.11 - Path Traversal [155]

Οι εξυπηρετητές χρησιμοποιούν δύο μηχανισμούς ασφαλείας για να περιορίσουν την πρόσβαση του χρήστη. Ο πρώτος είναι ο κατάλογος ρίζας (Root Directory) και ο δεύτερος είναι οι λίστες ελέγχου πρόσβασης (Access Control Lists ή ACLs). Ο κατάλογος ρίζας (Root Directory) είναι ο κορυφαίος κατάλογος στο σύστημα αρχείων του εξυπηρετητή και χρησιμοποιείται για να εμποδίσει την πρόσβαση των χρηστών σε αρχεία και φακέλους που βρίσκονται έξω από αυτόν. Οι διαχειριστές, χρησιμοποιούν τις λίστες ελέγχου πρόσβασης (ACLs) προκειμένου να καθορίσουν τα δικαιώματα πρόσβασης του χρήστη καθώς και τα

προνόμια για την προβολή (view), τη τροποποίηση (modify) και την εκτέλεση (execute) αρχείων.

Η εν λόγω επίθεση μπορεί να οφείλεται τόσο στην ευπάθεια του **ακατάλληλου χειρισμού εισόδου (Improper Input Handling)** όσο και σε αδυναμίες στο λογισμικό του εξυπηρετητή. Για την ανίχνευση ευπαθειών που μπορούν να οδηγήσουν στην υλοποίηση αυτής της επίθεσης οι προγραμματιστές διαδικτυακών εφαρμογών χρησιμοποιούν αντίστοιχους σαρωτές και εφαρμόζουν χειροκίνητα τεχνικές δοκιμών διείσδυσης. Επιπροσθέτως, θα πρέπει να ενημερώνουν ανά τακτά χρονικά διαστήματα το λογισμικό του εξυπηρετητή. [94] [95] [96] [97] [98]

2.2.32 **Προβλέψιμη τοποθεσία πόρων** (Predictable Resource Location)

Η προβλέψιμη τοποθεσία πόρων είναι μια τεχνική επίθεσης που χρησιμοποιείται για τον εντοπισμό κρυφού περιεχομένου και λειτουργιών μιας ιστοσελίδας. Λειτουργεί με τον ίδιο τρόπο που λειτουργεί μια επίθεση ωμής βίας (brute force attack) και αναζητά περιεχόμενο που δεν διατίθεται για δημόσια προβολή. Μερικά παραδείγματα τέτοιου περιεχόμενου είναι:

- ✓ Τα αρχεία καταγραφής (Logs Files)
- ✓ Τα προσωρινά αρχεία (Temporary Files)
- ✓ Τα αρχεία διαμόρφωσης (Configuration Files)
- ✓ Τα αρχεία ασφαλείας (Backup files)

Αυτές οι αναζητήσεις είναι εύκολες λόγω του ότι τα κρυφά αρχεία συνήθως έχουν συνηθισμένα ονόματα και βρίσκονται σε συγκεκριμένα σημεία. Αυτά τα αρχεία πιθανόν να εμπεριέχουν ευαίσθητες πληροφορίες σχετικά με την ιστοσελίδα, τις λειτουργίες της, την Βάση Δεδομένων, τα ονόματα συσκευών, τις διαδρομές αρχείων προς άλλες ευαίσθητες περιοχές κ.α. Επιπροσθέτως, η εν λόγω επίθεση μπορεί να αποκαλύψει στον εισβολέα πολύτιμες πληροφορίες σχετικά με το περιβάλλον και τους χρήστες του.

Η επίθεση αυτή είναι γνωστή ως καταναγκαστική περιήγηση (Forced Browsing), βίαιη περιήγηση (Forceful Browsing), καταμέτρηση αρχείων (File Enumeration) και καταμέτρηση καταλόγων (Directory Enumeration).

Το 8% των επιθέσεων που έχουν καταγραφεί στηρίζονται σε αυτή την επίθεση. Για την προστασία από αυτές τις επιθέσεις, θα πρέπει να γίνουν αλλαγές στις προεπιλεγμένες ρυθμίσεις του εργαλείου που διαχειρίζεται τις Βάσεις Δεδομένων (π.χ. του phpMyAdmin) και παράλληλα τα ονόματα των αρχείων, των ονομάτων χρηστών (usernames), των στηλών (στους πίνακες) των βάσεων δεδομένων καθώς και των καταλόγων θα πρέπει να μην είναι προφανή. [101] [102] [103]

2.2.33 **Συμπερίληψη απομακρυσμένου αρχείου** (Remote File Inclusion)

Η συμπερίληψη απομακρυσμένου αρχείου είναι μια επίθεση που δίνει την δυνατότητα στον εισβολέα να αποθηκεύσει στον εξυπηρετητή της εφαρμογής ένα αρχείο που μπορεί να επηρεάσει αρνητικά τον τρόπο με τον οποίο λειτουργεί η εφαρμογή. Τέτοια αρχεία συνήθως περιλαμβάνουν scripts σε JavaScript ή σε άλλες γλώσσες προγραμματισμού. Το γεγονός αυτό, συνήθως οφείλεται σε ακατάλληλο χειρισμό εισόδου και μπορεί να προκαλέσει προβλήματα στον εξυπηρετητή και στον χρήστη (client). Μια τέτοια επίθεση μπορεί να οδηγήσει σε:

- Σε υποκλοπή δεδομένων,
- σε **άρνηση υπηρεσίας (Denial of Service)**,
- σε **Cross Site Scripting (XSS)** επίθεση

Ωστόσο, υπάρχουν περιπτώσεις που το κακόβουλο αρχείο δεν είναι στον ίδιο εξυπηρετητή με την εφαρμογή αλλά ο προγραμματιστής έχει συνδέει με αυτήν και το θεωρεί αξιόπιστο. Πολλές φορές όμως, για διάφορους λόγους, το απομακρυσμένο αρχείο μπορεί να τροποποιηθεί με τέτοιο τρόπο που να προκαλέσει προβλήματα στην εφαρμογή. Σε αυτή την περίπτωση θα πρέπει να ρυθμιστεί σωστά ο server. Αν για παράδειγμα, πρόκειται για μια php εφαρμογή που φιλοξενείται σε apache εξυπηρετητή θα πρέπει, στο αρχείο php.ini, να ελεγχθούν οι ρυθμίσεις στα παρακάτω flags:

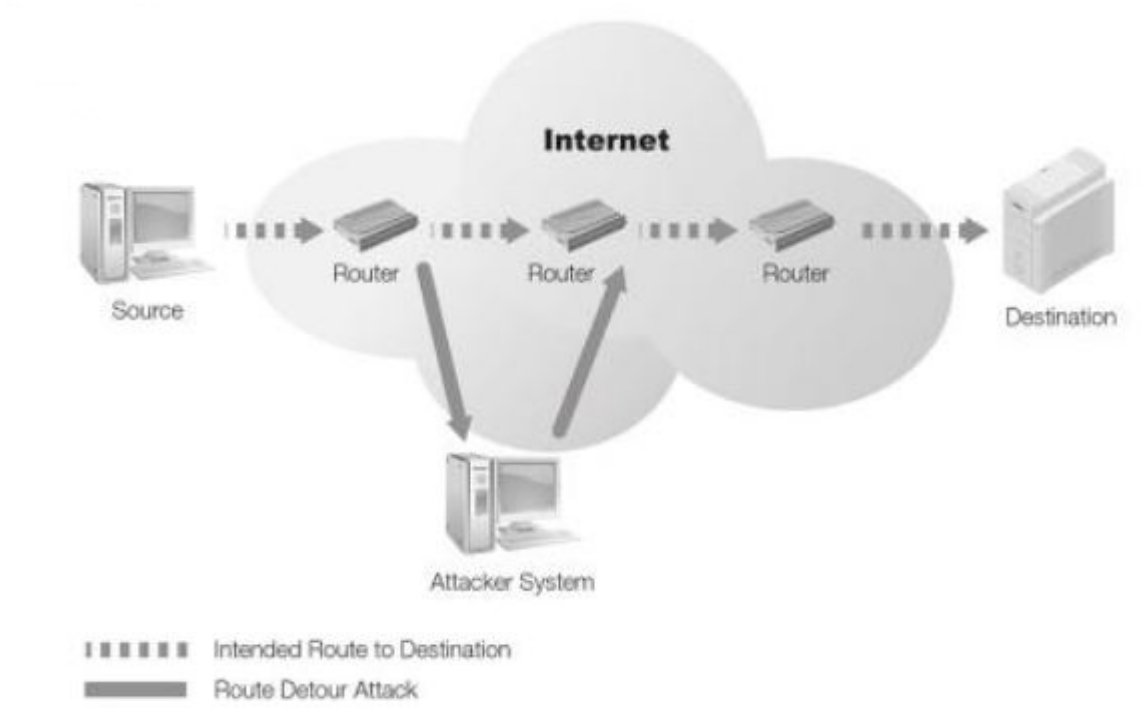
- ✓ allow_url_fopen - Υποδεικνύει αν τα εξωτερικά αρχεία μπορούν να συμπεριληφθούν. Η προεπιλεγμένη επιλογή είναι ορισμένη σε 'on' αλλά θα πρέπει να οριστεί σε 'off'.
- ✓ allow_url_include - Υποδεικνύει αν οι συναρτήσεις include(), require(), include_once(), και require_once() μπορούν να παραπέμπουν σε εξωτερικά αρχεία. Η προεπιλογή είναι στο 'off' και η allow_url_fopen είναι σε 'off' θα τεθεί και αυτή σε 'off'. [104]

2.2.34 **Παράκαμψη δρομολόγησης (**Routing Detour)

Η παράκαμψη δρομολόγησης είναι μια επίθεση του τύπου **Man-in-the-middle (MITM)** κατά την οποία ένας εισβολέας δρομολογεί εκ νέου τα δεδομένα σε μια εναλλακτική τοποθεσία. Η δρομολόγηση (routing) είναι μια σημαντική έννοια για τα δίκτυα υπολογιστών και το διαδίκτυο. Όλες οι online συναλλαγές δρομολογούνται σε έναν προορισμό, με βάση τις πληροφορίες που βρίσκονται εντός της επικεφαλίδας HTTP. Ένα μήνυμα διασχίζει το διαδίκτυο από τον έναν δρομολογητή (router) στον άλλον μέχρι να καταλήξει στον προορισμό του.

Το σύστημα δρομολόγησης που χρησιμοποιείται στο διαδίκτυο ενδεχομένως να έχει κενά ασφαλείας τα οποία μπορεί να εκμεταλλευτεί ένας εισβολέας προκειμένου να αλλάξει το προορισμό των δεδομένων. Σε ορισμένες περιπτώσεις, μπορεί τα δεδομένα να έχουν περάσει από έναν κακόβουλο προορισμό πριν καταλήξουν στον προορισμό που έπρεπε, χωρίς να γίνει αντιληπτή η υποκλοπή τους. Κάτι τέτοιο μπορεί να προκαλέσει σημαντικά προβλήματα, ειδικά

αν τα δεδομένα είναι ευαίσθητα, όπως πληροφορίες σχετικά με τραπεζικές συναλλαγές ή κωδικούς πρόσβασης. Η παρακάτω εικόνα δείχνει πως μια επίθεση παράκαμψης δρομολόγησης μπορεί να συμβεί.



Εικόνα 2.12 - Routing Detour [105]

Για την προστασία από αυτή την επίθεση θα πρέπει κατά τη σχεδίαση της εφαρμογής να καθοριστεί ένας ελάχιστος αριθμός ενδιάμεσων συστημάτων για τις αιτήσεις καθώς και ο απαραίτητος αριθμός SSL συνδέσεων με αμοιβαίο έλεγχο πιστοποίησης. [105]

2.2.35 **Εκμετάλλευση πινάκων σε μηνύματα** SOAP (SOAP Array Abuse)

Οι πίνακες XML του SOAP (Simple Object Access Protocol) είναι συνηθισμένος στόχος προκειμένου να επιτευχθεί κακόβουλη κατάχρηση. Οι πίνακες SOAP έχουν μια ή περισσότερες διαστάσεις (Rank) των οποίων τα μέλη βρίσκονται σε συγκεκριμένη θέση. Μια τιμή του πίνακα περιγράφεται ως μια σειρά από στοιχεία τα οποία απεικονίζονται στον πίνακα, με τα μέλη να εμφανίζονται σε αύξουσα σειρά. Όταν πρόκειται για πολυδιάστατους πίνακες, η τελευταία (προς τα δεξιά) διάσταση είναι αυτή που μεταβάλλεται ταχύτερα. Μια υπηρεσία διαδικτύου, η οποία περιμένει ένα τέτοιο πίνακα, μπορεί να είναι ο στόχος μιας XML DoS επίθεσης ωθώντας τον εξυπηρετητή SOAP να κατασκευάσει ένα τεράστιο πίνακα στη μνήμη της συσκευής, έτσι ώστε να προκαλέσει μια κατάσταση άρνησης υπηρεσίας (Denial of Service) στη συσκευή εξαιτίας της προ-δέσμευσης της μνήμης. [7]

2.2.36 Έγχυση σε αρχεία συμπερίληψης εξυπηρετητή (SSI Injection)

Η έγχυση σε αρχεία συμπερίληψης εξυπηρετητή (Server Side Includes ή SSI) είναι μια μορφή επίθεσης που επιτρέπει στον εισβολέα να στείλει κώδικα μέσα σε μια διαδικτυακή εφαρμογή, που στη συνέχεια θα εκτελεστεί τοπικά από τον εξυπηρετητή. Η επίθεση αυτή βασίζεται στην ευπάθεια του **ακατάλληλου χειρισμού εισόδου** των δεδομένων που πρόκειται να εισαχθούν σε ένα HTML αρχείο το οποίο διερμηνεύεται από τον εξυπηρετητή.

Οι εντολές συμπερίληψης αναλύονται και εκτελούνται από τον εξυπηρετητή πριν αποσταλεί μια HTML σελίδα και είναι χρήσιμες επειδή επιτρέπουν την αναπαραγωγή δυναμικού περιεχομένου σε μια υπάρχουσα σελίδα χωρίς να απαιτείται η εξ ολοκλήρου ανανέωσή της. Το χαρακτηριστικό αυτό είναι χρήσιμο σε πίνακες ανακοινώσεων, σε βιβλία επισκεπτών ή σε συστήματα διαχείρισης περιεχομένου καθώς και σε άλλες περιπτώσεις όπου οι χρήστες επιθυμούν να εισάγουν δεδομένα κατευθείαν στον κώδικα της ιστοσελίδας.

Αν ο εισβολέας είναι σε θέση να υποβάλλει μια εντολή συμπερίληψης αρχείου, τότε πιθανόν να έχει τη δυνατότητα να εκτελέσει αυθαίρετες εντολές στο λειτουργικό σύστημα, ή να συμπεριλάβει τα περιεχόμενα ενός περιορισμένου αρχείου, την επόμενη φορά που θα εμφανιστεί η σελίδα.

Για την προστασία από αυτή την επίθεση θα πρέπει να γίνουν οι κατάλληλες ρυθμίσεις στον εξυπηρετητή. Επιπροσθέτως, σε περίπτωση που η εν λόγω λειτουργία δεν είναι απαραίτητη, καλό θα ήταν να είναι απενεργοποιημένη. [7]

2.2.37 Κακόβουλος ορισμός αναγνωριστικών συνόδου (Session Fixation)

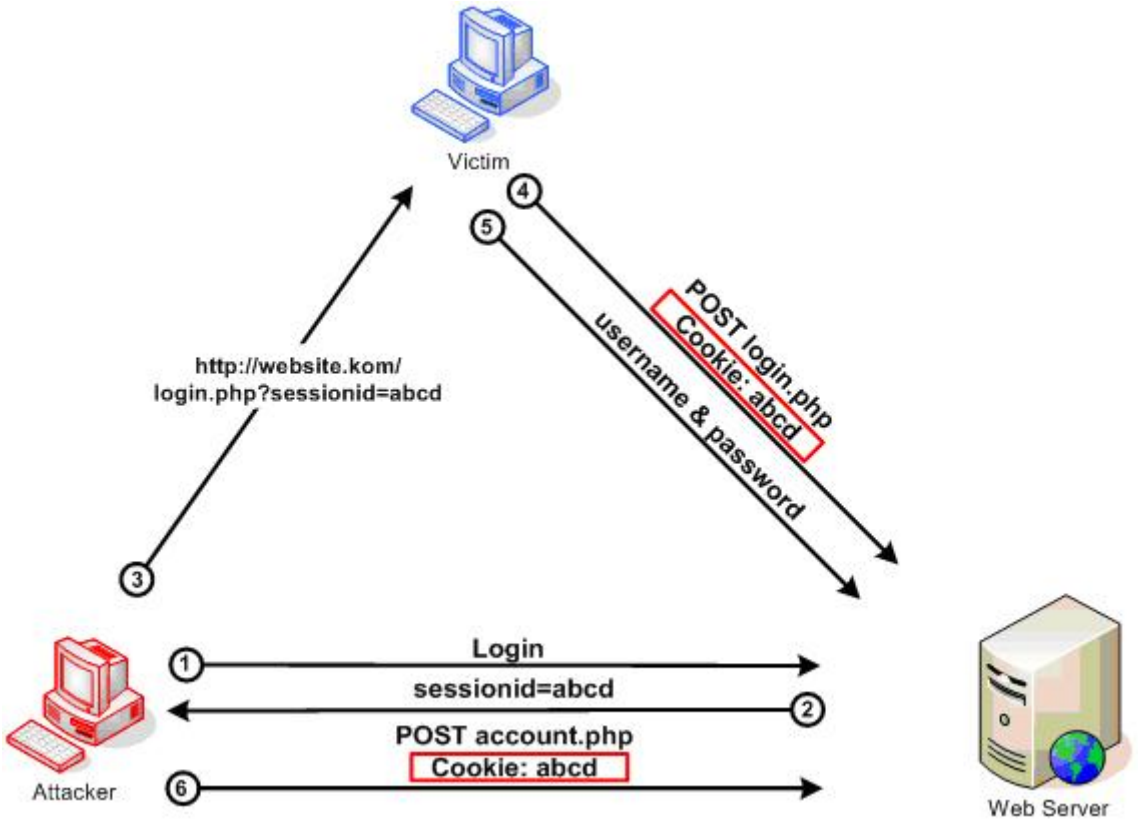
Ο κακόβουλος ορισμός αναγνωριστικών συνόδου είναι μια μορφή επίθεσης κατά την οποία ο επιτιθέμενος προσπαθεί να ορίσει το αναγνωριστικό συνόδου (session ID) ενός χρήστη. Σε αντίθεση με μια επίθεση **πειρατείας συνόδου (session hijacking)**, η εν λόγω επίθεση πραγματοποιείται πριν ο χρήστης συνδεθεί (login) με την ιστοσελίδα. Η εν λόγω επίθεση αποτελεί μια υποκατηγορία της επίθεσης πειρατείας συνόδου (session hijacking) και προσπαθεί να εκμεταλλευτεί τις αδυναμίες της εφαρμογής σε ότι αφορά τη διαχείριση των διακριτικών συνόδου (session tokens). Παρακάτω αναφέρονται μερικές από τις πιο συνηθισμένες τεχνικές εκμετάλλευσης:

- **Session token μέσα σε URL ορίσματα:** Όπου το session ID ορίζεται από τον επιτιθέμενο και στέλνεται στο θύμα μέσω mail, με τη μορφή μιας διεύθυνση URL.
- **Session token μέσα σε κρυφό πεδίο (σε μια φόρμα):** Όπου το θύμα μπορεί να εξαπατηθεί με το να συμπληρώσει τα διαπιστευτήρια του (username & password) σε μια φόρμα που μπορεί να βρίσκεται σε έναν άλλο δικτυακό τόπο, ο οποίος ανήκει στον επιτιθέμενο ή μέσα σε μια html φόρμα που έχει αποσταλεί μέσω mail.
- **Session ID μέσα σε cookies:** Οι συνεδρίες που βασίζονται σε cookies είναι και οι πιο εύκολοι στόχοι για να δεχθούν επίθεση. Τα περισσότερα σενάρια εκμετάλλευσης αποσκοπούν στη ρύθμιση (fixation) των cookies.

- Με επιθέσεις που συνδυάζουν **Cross Site Scripting (XSS)** τεχνικές και παλαιά HTTP αιτήματα (requests).
- Με επιθέσεις που βασίζονται στην έγχυση κώδικα στα **<meta> tags**. Η επιθέσεις αυτές είναι πιο αποτελεσματικές από τις επιθέσεις Cross-Site Scripting (XSS) επειδή δεν μπορούν να αντιμετωπιστούν με την ίδια ευκολία. Το γεγονός αυτό οφείλεται στο ότι είναι αδύνατο να απενεργοποιηθεί η επεξεργασία αυτών των ετικετών (tags) στα προγράμματα περιήγησης.

Εκτός από τις παραπάνω, υπάρχουν και άλλες τεχνικές εκμετάλλευσης που έχουν σαν αποτέλεσμα μη εγκεκριμένες ενέργειες από την πρόσβαση / παραχώρηση δικαιωμάτων σε μη εξουσιοδοτημένους χρήστες (Cross-calation), όπου ο εισβολέας μπορεί να αποκτήσει με επιτυχία πρόσβαση σε λειτουργίες και δεδομένα που υπό κανονικές συνθήκες δεν θα έπρεπε.

Στο παράδειγμα που ακολουθεί, περιγράφεται ένα συνηθισμένο σενάριο εκμετάλλευσης αυτής της επίθεσης.



Εικόνα 2.13 - Session Fixation [108]

Ο εισβολέας επιχειρεί να συνδεθεί (login) νόμιμα με τον εξυπηρετητή (1), ο οποίος εκδίδει ένα session ID (2). Αν δε συμβεί αυτό, ο εισβολέας μπορεί να δημιουργήσει μια νέα συνεδρία (session) με το προτεινόμενο session ID. Στην συνέχεια, ο εισβολέας θα πρέπει να στείλει στο θύμα ένα σύνδεσμο (URL) με το καθορισμένο (fixed) session ID (3). Όταν το θύμα επισκεφτεί το σύνδεσμο αυτό, ο εξυπηρετητής (4) θα λάβει τη σύνοδο (session), και δε θα χρειαστεί να ορίσει καινούρια. Στη συνέχεια, το θύμα θα εισάγει τα διαπιστευτήρια του (username & password) στον εξυπηρετητή (5). Από τη στιγμή που ο εισβολέας γνωρίζει το session ID, θα μπορέσει να αποκτήσει πρόσβαση στο λογαριασμό του χρήστη (6).

Για τη προστασία από αυτή την επίθεση απαιτείται ένας συνδυασμός διαφόρων αντίμετρων. Η μεθοδολογία αυτή είναι γνωστή με την ονομασία "Άμυνα στρατηγικής βάθους" (Defense in Depth) και υποστηρίζει ότι αν ένα εμπόδιο είναι ασήμαντο και μπορεί να ξεπεραστεί, τότε τα πολλά εμπόδια μπορούν να ξεπεραστούν με πολύ μεγαλύτερη δυσκολία. Συνεπώς, η άμυνα στρατηγικής βάθους, ανεξάρτητα από την γλώσσα προγραμματισμού που χρησιμοποιείται, θα μπορούσε να περιλαμβάνει:

- Τη χρήση SSL / TLS πρωτόκολλου ασφαλείας.
- Την αναπαραγωγή της τιμής του session ID όχι μόνο αμέσως μετά την σύνδεση (login) του χρήστη αλλά και κάθε φορά που αλλάζει το επίπεδο των δικαιωμάτων του. Με αυτό τον τρόπο ο εισβολέας δε θα είναι σε θέση να γνωρίζει τη νέα τιμή του session ID και η σύνοδος θα είναι ασφαλής, τουλάχιστον από αυτή τη μορφή επίθεσης.
- Τη χρήση ενός μηχανισμού αποσύνδεσης (log-out) ο οποίος επιτρέπει στους χρήστες να υποδείξουν ότι μια σύνοδος (session) δεν είναι πλέον ενεργή και δεν επιτρέπει περαιτέρω αιτήματα (requests).
- Τη χρήση ενός χρονομετρητή λήξης συνεδρίας (Time-out old session IDs).
- Την αποδοχή των session ID τα οποία έχουν παραχθεί μόνο από την πλευρά του εξυπηρετητή.

Εκτός από τα παραπάνω, ένας προγραμματιστής διαδικτυακών εφαρμογών οφείλει να λαμβάνει υπόψη του ότι:

- ✓ Δεν θα πρέπει να γίνονται δεκτά τα session ID από URL και από GET / POST μεταβλητές.
- ✓ Θα πρέπει να χρησιμοποιεί σωστά τα subdomains έτσι ώστε να μην υπάρχει καμία σύγκρουση μεταξύ τους, σε ότι αφορά τα cookies.
- ✓ Η αποθήκευση ενός session ID σε ένα cookie μπορεί να οδηγήσει στην υλοποίηση μιας επίθεσης πλαστογράφησης αιτήσεων μεταξύ ιστοχώρων (Cross Site Request Forgery)
- ✓ Το να χρησιμοποιείται η διεύθυνση IP του χρήστη, είναι ένας τρόπος να εξασφαλιστεί ότι ο χρήστης που εμφανίζεται στην αρχή είναι ο ίδιος με αυτόν στο τέλος. Αυτό σημαίνει ότι καθ όλη την διάρκεια μιας συνόδου η διεύθυνση IP θα πρέπει να είναι η ίδια. Ωστόσο, η μέθοδος αυτή δεν είναι αποτελεσματική όταν π.χ. αρκετοί χρήστες μοιράζονται την ίδια διεύθυνση IP.

- ✓ Οι περιηγητές αναγνωρίζουν τους εαυτούς τους με τις "User-Agent" HTTP επικεφαλίδες (headers). Μια επικεφαλίδα συνήθως δεν αλλάζει κατά την διάρκεια χρήσης του περιηγητή, ενώ θα ήταν εξαιρετικά ύποπτο αν επρόκειτο να αλλάξει. Μια διαδικτυακή εφαρμογή πιθανόν να έκανε χρήση μιας User-Agent ανίχνευσης στην προσπάθεια της να εμποδίσει κακόβουλους χρήστες να κλέψουν συνεδρίες (sessions). Ωστόσο, υπάρχουν αρκετά πράγματα που πρέπει να συνυπολογίσει ένας προγραμματιστής για τη σωστή χρήση αυτής της προσέγγισης.

- ✓ Η αναπαραγωγή ενός νέου session ID μετά από κάθε αίτημα (request) δεν είναι πάντα δυνατή. Το γεγονός αυτό συνήθως οφείλεται στην παράλληλη λειτουργία ενός third-party λογισμικού (π.χ. ActiveX) ή ενός πρόσθετου (plug-in) που βρίσκεται στον περιηγητή του χρήστη. Αυτό μπορεί να επηρεάσει την ομαλή λειτουργία της εφαρμογής είτε με την διάσπαση μιας συνόδου (session) σε δυο ξεχωριστές είτε με το να προκαλέσει αποσύνδεση (log out). Επιπλέον, αν η εφαρμογή των συνόδων εμπεριέχει τη μετάδοση του session ID μέσω GET ή POST μεταβλητών, τότε αυτό θα μπορούσε να καταστήσει το "πίσω" κουμπί στους περισσότερους περιηγητές άχρηστο, αφού θα είχε σαν αποτέλεσμα την ακύρωση των session ID. [106] [107] [108]

2.2.38 **"Δηλητηρίαση"** Cookie (Cookie Poisoning)

Μια επίθεση Cookie Poisoning έχει σαν στόχο τη τροποποίηση του περιεχομένου ενός cookie έτσι ώστε να παρακάμψει τους μηχανισμούς ασφαλείας μιας εφαρμογής διαδικτύου και να επιτρέψει στον εισβολέα να κλέψει την ταυτότητα ή άλλες πληροφορίες ενός χρήστη. Τέτοιες πληροφορίες μπορεί να είναι ο αριθμός μιας πιστωτικής κάρτας, ένα αναγνωριστικό συνόδου (Session ID) και άλλες πληροφορίες σχετικά με συναλλαγές. Πολλές φορές η εν λόγω επίθεση λέγεται και Session Poisoning, Session Pollution ή Session Modification.

Η εν λόγω επίθεση συνήθως οφείλεται στην ευπάθεια του **ακατάλληλου χειρισμού εισόδου** των δεδομένων που εισάγει ο χρήστης, τα οποία ενδεχομένως αντιγράφονται σε μεταβλητές συνόδου.

Άλλες φορές, μπορεί να είναι αποτέλεσμα κακής διαχείρισης των συνόδων (sessions) εντός της εφαρμογής. Κάτι τέτοιο μπορεί να οφείλεται είτε σε προγραμματιστικά λάθη είτε σε εσφαλμένη διαμόρφωση των αρχείων που βρίσκονται στον εξυπηρετητή (π.χ. πολλές φορές οι προγραμματιστές PHP εφαρμογών παροτρύνονται να θέσουν σε “off” την τιμή της παραμέτρου register_globals που βρίσκεται μέσα στο αρχείο php.ini).

Επιπροσθέτως, η επίθεση αυτή μπορεί να πραγματοποιηθεί όταν ο εισβολέας εισάγει κακόβουλα scripts μέσα στο περιβάλλον του εξυπηρετητή, πράγμα που είναι πιθανό να συμβεί μόνο όταν χρήστης και εισβολέας μοιράζονται τον ίδιο εξυπηρετητή. [10] [109] [110] [111] [112]

2.2.39 **Έγχυση σε εντολές** SQL (SQL Injection)

Η έγχυση σε εντολές SQL (Structure Query Language Injection ή SQLI) είναι μια επίθεση που οφείλεται κατά κύριο λόγο στην ευπάθεια ακατάλληλου χειρισμού εισόδου και εξόδου και έχει ως

αποτέλεσμα την εκτέλεση SQL εντολών (statements) που δεν έχουν προβλεφθεί από τον προγραμματιστή. Αυτές η εντολές μπορεί να έχουν σαν αποτέλεσμα:

- ✓ την προβολή (SELECT) δεδομένων που δεν θα έπρεπε να προβάλλονται στους χρήστες,
- ✓ την ανεξέλεγκτη εισαγωγή (INSERT) δεδομένων και
- ✓ τη διαγραφή (DELETE) όχι μόνο των δεδομένων αλλά και ολόκληρης της βάσης ή κάποιου πίνακα (table).

Τα SQLI εμφανίστηκαν πρώτη φορά το 1998. Είναι γνωστά ως επιθέσεις μόλυνσης (vector) για τις ιστοσελίδες και μπορούν να στραφούν εναντίων κάθε τύπου SQL βάσης δεδομένων. Ο OWASP έχει συμπεριλάβει αυτή την επίθεση μέσα στις κορυφαίες 10 επιθέσεις διαδικτυακών εφαρμογών από το 2007 ενώ το 2013, η εν λόγω επίθεση κατείχε την πρώτη θέση.

Τα αποτελέσματα τέτοιων επιθέσεων μπορεί να εμφανιστούν άμεσα στην οθόνη, πράγμα που σημαίνει ότι έχουμε ένα κλασικό (classic) SQLI. Ωστόσο, υπάρχουν και τα "τυφλά" (blind) SQLI τα αποτελέσματα των οποίων δεν είναι ορατά στον επιτιθέμενο. Η σελίδα με την αδυναμία πιθανόν να μην είναι μία από αυτές που εμφανίζουν δεδομένα αλλά μπορεί να εμφανιστεί διαφορετικά, αφού βασίζεται στα αποτελέσματα μιας λογικής SQL εντολής (statement) η οποία έχει αλλάξει με αθέμιτο τρόπο. Αυτός ο τύπος επίθεσης μπορεί να είναι χρονοβόρος, επειδή μια νέα εντολή πρέπει να δημιουργείται για κάθε μια ανάκτηση. Υπάρχουν αρκετά εργαλεία που μπορούν να αυτοματοποιήσουν αυτές τις επιθέσεις μόλις συσταθούν η τοποθεσία της αδυναμίας και ο στόχος των πληροφοριών.

Μια Second Order SQLI συμβαίνει όταν υποβάλλονται δεδομένα που εμπεριέχουν λανθασμένες SQL εντολές και αντί να εκτελεστούν αμέσως, αποθηκεύονται στην ΒΔ. Σε μερικές περιπτώσεις, η εφαρμογή μπορεί να έχει κωδικοποιήσει σωστά μια SQL εντολή και να την έχει αποθηκεύσει σαν αξιόπιστη. Στην συνέχεια, ένα άλλο μέρος αυτής της εφαρμογής, χωρίς να κάνει τους κατάλληλους ελέγχους μπορεί να εκτελέσει αυτή την αποθηκευμένη εντολή. Αυτή η επίθεση απαιτεί περισσότερη γνώση για το πως χρησιμοποιούνται τα δεδομένα μετά την υποβολή τους. Οι αυτοματοποιημένοι σαρωτές διαδικτυακών εφαρμογών δεν θα μπορέσουν να εντοπίσουν εύκολα αυτό το είδος SQLI και πιθανόν να πρέπει να γίνουν έλεγχοι χειροκίνητα.

Τα SQLI μπορούν να συνδυαστούν με την ευπάθεια της ανεπαρκούς αυθεντικοποίησης αλλά και με τις παρακάτω επιθέσεις:

- ➢ Κατανεμημένες επιθέσεις άρνησης υπηρεσίας (DDoS)
- ➢ Cross-Site Scripting
- ➢ DNS Hijacking

Το ευχάριστο με τα SQLI είναι ότι μπορούν εύκολα να αντιμετωπιστούν ανεξάρτητα από την γλώσσα προγραμματισμού που χρησιμοποιείται. Για την πρόληψη αυτής της επίθεσης θα πρέπει πρώτα από όλα ο προγραμματιστής να είναι υποψιασμένος ως προς οποιαδήποτε καταχώρηση (input) από το χρήστη η οποία μπορεί να γίνει είτε από κάποια φόρμα είτε από το URL. Για αυτό το λόγο θα πρέπει κατά την εισαγωγή να γίνεται επικύρωση (validating) ή καθαρισμός (sanitizing) των δεδομένων ενώ κατά την εξαγωγή τους (output) θα πρέπει να γίνεται διαφυγή των ακατάλληλων χαρακτήρων (escaping).

Εκτός από τα παραπάνω, ένας προγραμματιστής μπορεί να κάνει χρήση προκατασκευασμένων εντολών γνωστές ως Prepared Statements ή Parameterized Statements, οι οποίες λειτουργούν με παραμέτρους που μπορούν να χρησιμοποιηθούν για να ελέγχουν το περιεχόμενο του χρήστη (input) πριν την εκτέλεση μιας εντολής (statement). Αυτή η μέθοδος συνήθως χρησιμοποιεί σε συνδυασμό με κάποιο αυστηρό πρότυπο (pattern) βάση του όποιου μπορούν να ελέγχονται μεταβλητές που απαιτούν απλές τιμές όπως ακέραιες (integers), δεκαδικές (float) ή λογικές (boolean) αλλά και πιο σύνθετες όπως ημερομηνίες (dates), αριθμούς 128-bit (Universal Unique Identifier ή UUID), αλφαριθμητικές τιμές (alphanumeric) χωρίς ειδικούς χαρακτήρες κλπ. Ένα τέτοιο πρότυπο είναι το Active Record Pattern.

Επιπροσθέτως, περιορίζοντας τα δικαιώματα (permissions) σύνδεσης μεταξύ της βάσης δεδομένων και της διαδικτυακής εφαρμογής στο βαθμό που πρέπει, ίσως να βοηθούσε αποτελεσματικά στον μετριασμό απειλών για κάθε τύπου SQLI που εκμεταλλεύονται τυχόν σφάλματα (bugs) στη διαδικτυακή εφαρμογή. [113] [114]

2.2.40 **Κατάχρηση ανακατεύθυνσης** URL (URL Redirector Abuse)

Η λειτουργία ανακατεύθυνσης URL χρησιμοποιείται από τις διαδικτυακές εφαρμογές για να προωθήσει ένα εισερχόμενο αίτημα σε έναν εναλλακτικό πόρο. Η λειτουργία αυτή είναι απαραίτητη όταν ο χρήστης αναζήτα κάποιον πόρο ο οποίος έχει μετακινηθεί μέσα στην εφαρμογή και ο χρήστης τον αναζητά με την παλιά URL διεύθυνση. Αυτό έχει σαν αποτέλεσμα να εξυπηρετείται ο χρήστης και να διατηρεί η εφαρμογή την λειτουργικότητά της χωρίς να εμφανίζει μηνύματα σφάλματος. Άλλες φορές, η εν λόγω λειτουργία χρησιμοποιείται για την εξισορρόπηση φορτίου, αξιοποιώντας συντετμημένες URL διευθύνσεις ή για την καταγραφή εξερχόμενων συνδέσμων. Δυστυχώς, η τελευταία χρηστικότητα μπορεί να δώσει την δυνατότητα σε έναν εισβολέα να υλοποιήσει μια επίθεση **phishing**.

Αυτό που μπορεί να κάνει ένας προγραμματιστής για να αποφύγει αυτή την επίθεση είναι να μην επιτρέπει σε έναν χρήστη να εισάγει URL διευθύνσεις ή όταν του επιτρέπει να κάνει κάτι τέτοιο θα πρέπει να καταγράφει αυτές τις URL διευθύνσεις και να χρησιμοποιεί μια λίστα με διευθύνσεις που θεωρεί έμπιστες ή κακόβουλες. Εναλλακτικά, μπορεί να επιτρέπει σε έναν χρήστη να εισάγει μια URL διεύθυνση, η οποία πρώτα θα επικυρώνεται από τον διαχειριστή της εφαρμογής και μετά θα χρησιμοποιείται για ανακατεύθυνση.

Ο OWASP αναφέρει την εν λόγω επίθεση με τίτλο "Μη επικυρωμένες ανακατευθύνσεις και προωθήσεις" (Unvalidated Redirects and Forwards). [7]

2.2.41 **Έγχυση σε** Xpath (Xpath Injection)

Όπως μια επίθεση έγχυσης SQL, έτσι και μια επίθεση έγχυσης Xpath πραγματοποιείται όταν μια ιστοσελίδα χρησιμοποιεί τα δεδομένα που εισάγει ο χρήστης για να κατασκευάσει ένα ερώτημα (query) Xpath για XML δεδομένα. Με την εισαγωγή κακόβουλων δεδομένων μέσα στην εφαρμογή, ο εισβολέας μπορεί να μάθει με ποιο τρόπο είναι δομημένος ο XML κώδικας, ή να αποκτήσει πρόσβαση σε δεδομένα που δεν θα έπρεπε. Μπορεί ακόμη και να είναι σε θέση να αυξήσει τα προνόμιά του (privileges) στην εφαρμογή, αν τα XML δεδομένα χρησιμοποιούνται για τον έλεγχο ταυτότητας.

Η υποβολή ερωτημάτων (queries) XML γίνεται με την χρήση της γλώσσας Xpath η οποία χρησιμοποιεί συγκεκριμένα χαρακτηριστικά και πρότυπα, όπως γίνεται και με την γλώσσα SQL. Συνήθως, μια εφαρμογή συνδυάζει τον XML κώδικα με τα δεδομένα που εισάγει ένας χρήστης προκειμένου να διαμορφώσει το περιεχόμενο που θα εμφανιστεί στην οθόνη. Για το λόγο αυτό θα πρέπει να λαμβάνονται μέτρα για την σωστή διαχείριση των δεδομένων που εισάγονται από τους χρήστες πριν χρησιμοποιηθούν σε ερωτήματα Xpath.

Η Xpath είναι μα τυποποιημένη γλώσσα, αυτό σημαίνει ότι η σύνταξή της είναι πάντα ανεξάρτητη από τον τρόπο υλοποίησης της εφαρμογής και δεν υπάρχουν περιθώρια αλλαγής από τη στιγμή που χρησιμοποιείται σε αιτήματα (requests) προς την βάση δεδομένων. Αν η εφαρμογή δεν προστατεύεται από την ευπάθεια του ακατάλληλου χειρισμού εισόδου και εξόδου μπορεί να επιτρέψει την υλοποίηση αυτοματοποιημένων επιθέσεων. [115]

2.2.42 **Καταιγισμός γνωρισμάτων** XML (XML Attribute Blowup)

Μια επίθεση καταιγισμού γνωρισμάτων XML πραγματοποιείται όταν ο εισβολέας διοχετεύει ένα κακόβουλο XML αρχείο προκειμένου να επηρεάσει την ομαλή λειτουργία ενός συντακτικού αναλυτή XML. Ένας ευάλωτος συντακτικός αναλυτής XML, θα διαχειριστεί με λάθος τρόπο τα πολυάριθμα γνωρίσματα (attributes) που περιλαμβάνει το κακόβουλο αρχείο XML, στη συνέχεια θα επιβαρύνει τον επεξεργαστή και θα οδηγήσει σε άρνηση υπηρεσίας (Denial of Service).

Για την αποφυγή τέτοιων επιθέσεων θα πρέπει να εφαρμόζεται μια αυστηρή μέθοδος (schema) επικύρωσης όλων των εισερχόμενων XML αρχείων. Η διαδικασία επικύρωσης θα πρέπει να επιβάλει κάποια όρια σχετικά με το μέγιστο μέγεθος ενός πίνακα (array) (ως προς τον αριθμό των γραμμών και των στηλών), το μέγιστο αριθμό των στοιχείων (elements), το μέγιστο αριθμό των γνωρισμάτων (attributes) ανά στοιχείο (element) κλπ. [116]

2.2.43 **Εξωτερικές οντότητες** XML (XML External Entities)

Η χρήση ενός συντακτικού αναλυτή XML (XML parser) ο οποίος έχει ρυθμιστεί ώστε να μην εμποδίζει ούτε να περιορίζει την ανάλυση των εξωτερικών οντοτήτων μπορεί να προκαλέσει μια επίθεση XML External Entities (XXE).

Οι XXE επιθέσεις επωφελούνται από ένα χαρακτηριστικό της XML το οποίο επιτρέπει να κατασκευαστούν αρχεία, με δυναμικό τρόπο, κατά τον χρόνο της επεξεργασίας. Μια XML οντότητα επιτρέπει την συμπερίληψη δεδομένων, με δυναμικό τρόπο, από μια δεδομένη πηγή. Οι εξωτερικές οντότητες επιτρέπουν σε ένα XML αρχείο να συμπεριλάβει δεδομένα από ένα εξωτερικό URI. Αν δεν έχουν γίνει οι κατάλληλες ρυθμίσεις, οι εξωτερικές οντότητες αναγκάζουν το συντακτικό αναλυτή XML να έχει πρόσβαση σε πόρους που ορίζονται από το URI, π.χ. σε κάποιο αρχείο που βρίσκεται είτε τοπικά στο μηχάνημα είτε σε κάποιο απομακρυσμένο σύστημα.

Όταν μια εφαρμογή είναι εκτεθειμένη σε αυτή την τεχνική επίθεσης, οι εξωτερικές οντότητες μπορεί να αντικαταστήσουν την κανονική τιμή της οντότητας με κακόβουλα δεδομένα, να εναλλάξουν παραπομπές, ακόμα και να θέσουν σε κίνδυνο την ασφάλεια των δεδομένων στα οποία έχει πρόσβαση η εφαρμογή XML ή και ο εξυπηρετητής. Επιπλέον, μπορεί να οδηγήσει σε επίθεση άρνησης υπηρεσίας (Denial of Service), να επιτρέψει τη σάρωση σε απομακρυσμένα μηχανήματα και τη μη εξουσιοδοτημένη πρόσβαση σε αρχεία που βρίσκονται στο μηχάνημα. [117]

2.2.44 **Ανάπτυξη οντοτήτων** XML (XML Entity Expansion)

Η XML επιτρέπει τη χρήση των DTDs (Document Type Definition). Τα DTDs έχουν ως στόχο να καθορίσουν την αναμενόμενη δομή ενός XML αρχείου. Μια από τις δυνατότητες των DTDs είναι να ορίζει οντότητες. Οι οντότητες είναι προσαρμοσμένες μακροεντολές που χρησιμοποιούνται για να ορίσουν συντομεύσεις για αλφαριθμητικούς ή ειδικούς χαρακτήρες. Τυπικά παραδείγματα προκαθορισμένων οντοτήτων είναι οι οντότητες που χρησιμοποιούνται εντός της HTML. Όταν για παράδειγμα πρέπει να χρησιμοποιηθούν χαρακτήρες όπως "<" ή ">" εκτός των HTML tags, θα πρέπει να αντικατασταθούν από τις οντότητες τους:

- ✓ ο χαρακτήρας "<" έχει την οντότητα ">"
- ✓ ο χαρακτήρας ">" έχει την οντότητα "<"

Οι οντότητες που δεν είναι προκαθορισμένες, μπορούν να δηλωθούν εσωτερικά ή εξωτερικά.

- ➢ Οι οντότητες που είναι δηλωμένες εσωτερικά, ορίζονται μέσα στο ίδιο αρχείο
- ➢ Οι οντότητες που είναι δηλωμένες εξωτερικά, ορίζονται σε κάποιο εξωτερικό αρχείο. Σε αυτή την περίπτωση δίνεται μόνο η διεύθυνση προς το εξωτερικό αρχείο.

Για την υλοποίηση μιας επίθεσης ανάπτυξης οντοτήτων XML, ο εισβολέας χρησιμοποιεί ένα έγγραφο XML που περιλαμβάνει ένα σύνολο προσαρμοσμένων οντοτήτων. Με αυτό τον τρόπο, ο εισβολέας μπορεί να υπερφορτώσει το συντακτικό αναλυτή (parser) XML για να εξαντλήσει τους διαθέσιμους πόρους του εξυπηρετητή και να προκαλέσει άρνηση υπηρεσίας (Denial of Service).

Αν ο κατασκευαστής έχει χρησιμοποιήσει σωστά το πρότυπο SOAP 1.2, τότε η εφαρμογή δεν θα είναι εκτεθειμένη σε αυτή την επίθεση. [118]

2.2.45 **Έγχυση** XML (XML Injection)

Κατά την διάρκεια μιας έγχυσης XML, ο επιτιθέμενος προσπαθεί να εισάγει διάφορα XML Tags σε μηνύματα SOAP (Simple Object Access Protocol) με στόχο να αλλάξει την δομή του XML κώδικα. Συνήθως, μια επιτυχημένη επίθεση έγχυσης XML προκαλείται κατά την εκτέλεση μιας απαγορευμένης λειτουργίας. Τέτοιες λειτουργίες μπορεί να έχουν ως αποτέλεσμα:

- Την μετατροπή των στοιχείων πληρωμής, πράγμα που σημαίνει παραβίαση της ακεραιότητας (Integrity)
- Την μη εξουσιοδοτημένη πρόσβαση με δικαιώματα (permissions) διαχειριστή (administrator), που παραβιάζει τον έλεγχο πρόσβασης.

Για να προστατευτεί μια εφαρμογή από αυτή την επίθεση θα πρέπει να ληφθούν μέτρα προστασίας για την ευπάθεια του ακατάλληλου χειρισμού εισόδου και εξόδου των δεδομένων που εισάγονται από το χρήστη. Σύμφωνα με τον WASC, ο προγραμματιστής θα πρέπει να είναι ιδιαίτερα προσεκτικός κατά το στάδιο της υλοποίησης της εφαρμογής. [119]

2.2.46 **Έγχυση** XQuery (XQuery Injection)

Η XQuery για την XML είναι ότι και η SQL για μια Βάση Δεδομένων. Κατά συνέπεια, η έγχυση XQuery λειτουργεί με την ίδια φιλοσοφία που λειτουργεί και μια επίθεση έγχυσης SQL (SQL Injection), με τη διαφορά ότι η πρώτη έχει να κάνει με XML δεδομένα.

Η έγχυση XQuery οφείλεται κατά κύριο λόγο στην ευπάθεια ακατάλληλου χειρισμού εισόδου και έχει ως αποτέλεσμα την εκτέλεση XQuery εντολών που δεν έχουν προβλεφθεί από τον προγραμματιστή. Η εν λόγω επίθεση μπορεί να χρησιμοποιηθεί για να απαριθμήσει στοιχεία στο περιβάλλον του θύματος, να εγχύσει εντολές τοπικά (local host), ή να εκτελέσει ερωτήματα (queries) σε απομακρυσμένα αρχεία και πηγές δεδομένων.

Για την αντιμετώπιση αυτής της επίθεσης θα πρέπει να γίνεται επαρκείς επικύρωση των δεδομένων εισόδου του χρήστη. Επιπλέον, θα πρέπει να χρησιμοποιούνται παραμετροποιημένα ερωτήματα (parameterized queries) και να δηλώνονται οι μεταβλητές που χρησιμοποιούνται. [120]

2.3 Μηχανισμοί Ασφάλειας

2.3.1 Κωδικός πρόσβασης

Ένας εύκολος και πολύ διαδεδομένος τρόπος για να πιστοποιήσει ένας χρήστης την ταυτότητά του είναι η χρήση διαπιστευτηρίων πρόσβασης. Τα διαπιστευτήρια ενός χρήστη είναι ο μοναδικός συνδυασμός ενός ονόματος (username) και ενός κωδικού πρόσβασης (password). Κάθε φορά που ένας χρήστης εισάγει τα διαπιστευτήριά του, η εφαρμογή ελέγχει αν είναι έγκυρα και στη συνέχεια παραχωρεί στο χρήστη τα αντίστοιχα δικαιώματα.

Σε λειτουργίες που σχετίζονται με ηλεκτρονικές συναλλαγές, οι εφαρμογές ζητούν από τους χρήστες να εισάγουν επιπλέον πληροφορίες / διαπιστευτήρια, όπως για παράδειγμα έναν δεύτερο κωδικό πρόσβασης ή να απαντήσουν σε μια ερώτηση ασφαλείας. Οι λειτουργίες αυτές μπορεί να απαιτούν από τους χρήστες επιπλέον διαδραστικότητα, πράγμα που μπορεί να προκαλεί δυσαρέσκεια αλλά ταυτόχρονα αποτελεί μια επιπρόσθετη δικλείδα ασφαλείας. [2]

2.3.2 Κρυπτογραφία

Ως **κρυπτογραφία** μπορεί να οριστεί ο επιστημονικός κλάδος που ασχολείται με την ασφαλή αποθήκευση της ευαίσθητης πληροφορίας, καθώς επίσης και με την αποστολή αυτής μέσω μη ασφαλών δικτύων, εξασφαλίζοντας την ανάγνωσή της μόνο από τον εξουσιοδοτούμενο παραλήπτη.

Η εφαρμογή της κρυπτογραφίας είναι η κρυπτογράφηση. **Κρυπτογράφηση (Encryption)** είναι ο μετασχηματισμός δεδομένων σε μορφή που να είναι αδύνατον να διαβαστεί χωρίς τη γνώση της σωστής ακολουθίας bit. Η ακολουθία bit καλείται "κλειδί" και είναι μια μυστική τιμή που χρησιμοποιεί ο αλγόριθμος κρυπτογράφησης για να κρυπτογραφήσει τα δεδομένα. Ο αλγόριθμος κρυπτογράφησης είναι μια μαθηματική συνάρτηση που χρησιμοποιείται στη διαδικασία της κρυπτογράφησης αλλά και στην διαδικασία της αποκρυπτογράφησης των δεδομένων. Η **αποκρυπτογράφηση (Decryption)** είναι η αντίστροφη διαδικασία της κρυπτογράφησης και απαιτεί γνώση του κλειδιού. Με αυτό τον τρόπο εξασφαλίζεται το απόρρητο των δεδομένων, κρατώντας τα κρυφά από όλους όσους έχουν πρόσβαση σε αυτά.

Τα συστήματα κρυπτογράφησης χωρίζονται σε συμμετρικά και ασύμετρα. Στα **συμμετρικά συστήματα ή συστήματα ιδιωτικού κλειδιού** χρησιμοποιείται το ίδιο κλειδί για κρυπτογράφηση και αποκρυπτογράφηση των δεδομένων. Στα **ασύμμετρα συστήματα ή κρυπτογραφία δημοσίου κλειδιού** χρησιμοποιείται ένα ζεύγος κλειδιών, το ιδιωτικό κλειδί και το δημόσιο κλειδί. Το δημόσιο κλειδί χρησιμοποιείται κατά την κρυπτογράφηση και το ιδιωτικό κατά την αποκρυπτογράφηση. [10] [2] [120]

Οι πιο γνωστοί αλγόριθμοι κρυπτογράφησης είναι οι εξής:

- Data Encryption Standard (DES)
- Triple DES

- RC2 και RC4
- International Data Encryption Algorithm (IDEA)
- RSA
- Diffie-Hellman.

2.3.3 **Ψηφιακές Υπογραφές** (Digital Signatures)

Οι ψηφιακές υπογραφές χρησιμοποιούνται για να πιστοποιούν την ταυτότητα του αποστολέα και την ακεραιότητα ενός μηνύματος. Βασίζονται στη χρήση της ασύμμετρης κρυπτογραφίας, πράγμα που σημαίνει ότι χρησιμοποιείται ένα ιδιωτικό κλειδί για την κρυπτογράφηση και ένα δημόσιο κλειδί για την αποκρυπτογράφηση ενός μηνύματος.

Όταν ένα ψηφιακό μήνυμα συνοδεύεται από μια ψηφιακή υπογραφή, ο παραλήπτης μπορεί να είναι σίγουρος για το ότι δεν έχει αλλοιωθεί. Μια ψηφιακή υπογραφή προστίθεται σε ένα μήνυμα όπως προστίθεται και η υπογραφή σε κάποιο έγγραφο προκειμένου να επιβεβαιώσει την αυθεντικότητα και τη μη αποποίηση ευθύνης του αποστολέα. [121] [122] [2]

Η λειτουργία των ψηφιακών υπογραφών έχει ως εξής:

1. Ο αποστολέας δημιουργεί το μήνυμα και στη συνέχεια χρησιμοποιεί μια συνάρτηση κατακερματισμού (hash function), για τη σύνοψή του.
2. Με το ιδιωτικό κλειδί του αποστολέα κρυπτογραφείται η σύνοψη του μηνύματος. Αυτή είναι και η ψηφιακή υπογραφή του αποστολέα.
3. Κρυπτογραφείται το αρχικό μήνυμα και η ψηφιακή υπογραφή με το δημόσιο κλειδί του παραλήπτη. Αυτός είναι ο ψηφιακός φάκελος του αποστολέα.
4. Ο αποστολέας στέλνει το ψηφιακό φάκελο στον παραλήπτη, ο οποίος τον αποκρυπτογραφεί με το ιδιωτικό του κλειδί. Έτσι ο παραλήπτης έχει αντίγραφο του μηνύματος και της ψηφιακής υπογραφής του αποστολέα.
5. Ο παραλήπτης με το δημόσιο κλειδί του αποστολέα αποκρυπτογραφεί την ψηφιακή υπογραφή του αποστολέα. Έτσι ο παραλήπτης έχει διαθέσιμη την αρχική σύνοψη του μηνύματος.
6. Ο παραλήπτης δημιουργεί μια σύνοψη του μηνύματος που παρέλαβε και τη συγκρίνει με τη σύνοψη που παρέλαβε από τον αποστολέα. Αν οι δυο συνόψεις είναι ίδιες βγαίνει το συμπέρασμα ότι το μήνυμα είναι αυθεντικό.

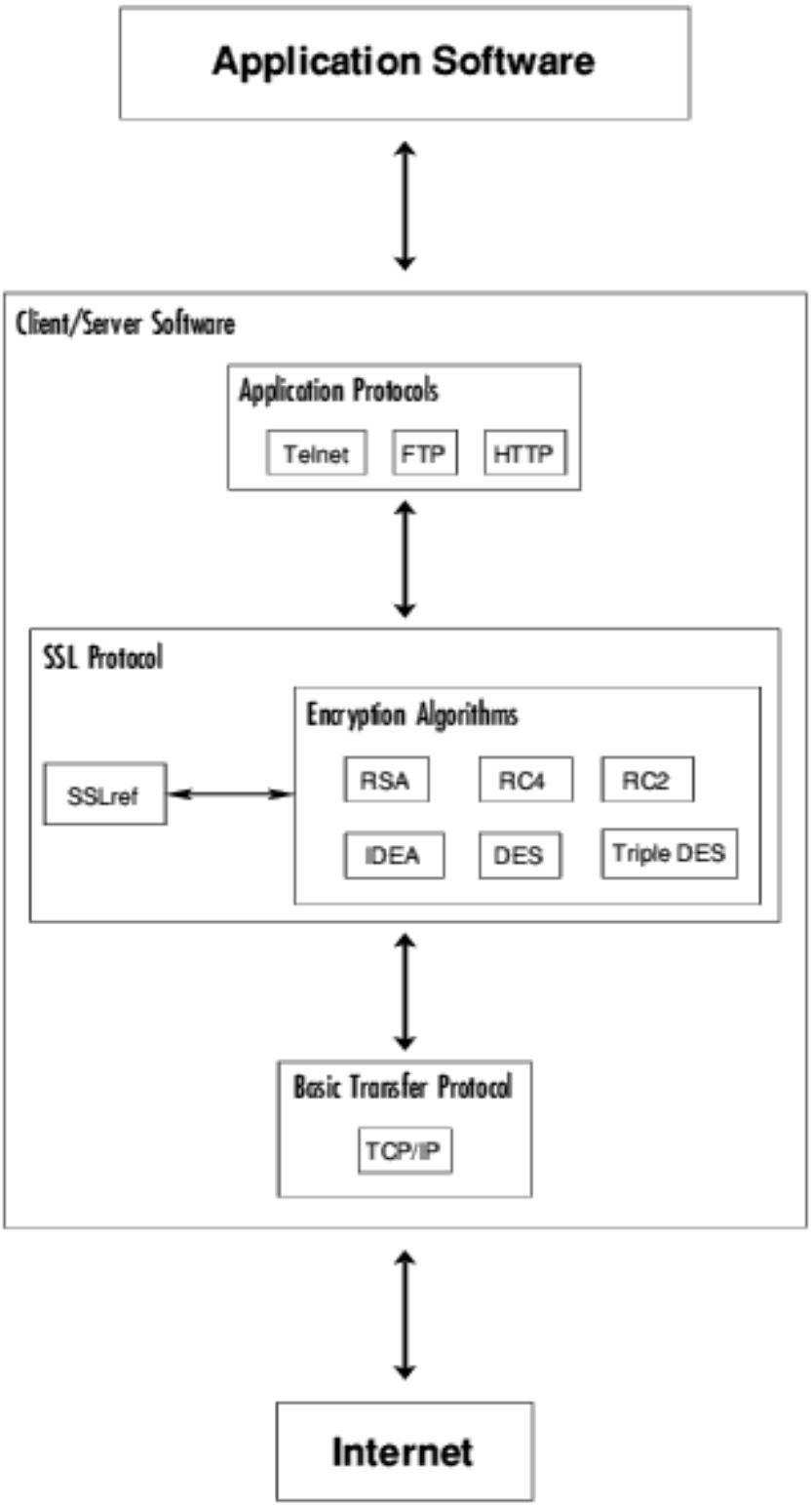
2.3.4 Secure Socket Layer (SSL)

Το πρωτόκολλο SSL (Secure Socket Layer) αναπτύχθηκε από την Netscape Communications Corporation για να παρέχει απόρρητη επικοινωνία μεταξύ δύο συστημάτων, εκ των οποίων το ένα λειτουργεί ως πελάτης (client) και το άλλο ως εξυπηρετητής (server). Χρησιμοποιείται από τον Ιούλιο του 1994 και σήμερα αποτελεί το κύριο πρωτόκολλο ασφαλείας στις ηλεκτρονικές συναλλαγές. Στα πλαίσια της εξέλιξής του, το Μάιο του 1996 πέρασε στα χέρια του IETF (Internet Engineering Task Force) και μετονομάστηκε σε TLS (Transport Layer Security), ωστόσο εξακολουθεί να είναι γνωστό ως SSL.

Το SSL παρέχει κρυπτογράφηση της μεταδιδόμενης πληροφορίας (Data Encryption), υποχρεωτική πιστοποίηση της ταυτότητας του εξυπηρετητή (Server Authentication) και προαιρετική πιστοποίηση της ταυτότητας του πελάτη (Client authentication) μέσω έγκυρων πιστοποιητικών που έχουν εκδοθεί από έμπιστες Αρχές Πιστοποίησης (Certificates Authorities). Υποστηρίζει πολλούς μηχανισμούς κρυπτογράφησης καλύπτοντας διαφορετικές ανάγκες. Τέλος, εξασφαλίζει την ακεραιότητα των δεδομένων (Data Integrity), εφαρμόζοντας την τεχνική των Message Authentication Codes (MACs), ώστε κανείς να μην μπορεί να αλλοιώσει την πληροφορία χωρίς να γίνει αντιληπτός. Φυσικά, όλα τα παραπάνω γίνονται με τρόπο διαφανές και απλό.

Το πρωτόκολλο αυτό αποτελείται από δύο επιμέρους πρωτόκολλα, το SSL record protocol και το SSL handshake protocol. Το SSL record protocol τοποθετεί τα δεδομένα σε πακέτα (packets), τα κρυπτογραφεί και τα μεταδίδει. Επιπλέον, εκτελεί την αντίστροφη διαδικασία για τα παραλαμβανόμενα πακέτα. Το SSL handshake protocol διαπραγματεύεται τους αλγόριθμους κρυπτογράφησης που θα χρησιμοποιηθούν και πραγματοποιεί την πιστοποίηση της ταυτότητας του εξυπηρετητή και του πελάτη (client) αν αυτό ζητηθεί. Μετά την ολοκλήρωση του SSL Handshake protocol, τα δεδομένα των εφαρμογών μπορούν να αποστέλλονται μέσω του SSL record protocol ακολουθώντας τις συμφωνημένες παραμέτρους ασφαλείας.

Το SSL μπορεί να τοποθετηθεί στην κορυφή οποιουδήποτε πρωτοκόλλου μεταφοράς, δεν εξαρτάται από την ύπαρξη του TCP/IP και τρέχει κάτω από πρωτόκολλα εφαρμογών όπως το HTTP, FTP και TELNET. Μια αναπαράσταση του πρωτοκόλλου SSL βλέπουμε παρακάτω. [123] [10]



Εικόνα 2.14 SSL [123]

Η παγκόσμια αγορά των αρχών έκδοσης πιστοποιητικών SSL, αποτελείται από ένα μικρό αριθμό οργανισμών και, σύμφωνα με έρευνα από την W3Techs, το μερίδιο αγοράς των πιο μεγάλων εταιριών πιστοποίησης κυμαίνεται στα παρακάτω επίπεδα:

- **Symantec**, με ποσοστό 38,1%
- **Comodo Group**, με ποσοστό 29,1%
- **Go Daddy**, με ποσοστό 13,4%
- **GlobalSign**, με ποσοστό 10%

Κάθε αρχή έκδοσης του πιστοποιητικού SSL έχει την υποχρέωση να επιβεβαιώσει τα στοιχεία της οντότητας, που αιτείται το πιστοποιητικό και ανάλογα με τα βήματα που ακολουθεί για την επιβεβαίωση, έχουν προκύψει τρεις βασικές κατηγορίες πιστοποιητικών SSL:

Το **Domain Validation** είναι ένα πιστοποιητικό βασικού επιπέδου. Το μόνο που χρειάζεται από αυτόν που αιτείται το συγκεκριμένο πιστοποιητικό είναι να επιβεβαιώσει ότι το domain name είναι έγκυρο και ανήκει στην ιδιοκτησία του. Δεν απαιτείται η κατάθεση κάποιου εγγράφου, αρκεί ένα απλό κλικ στον σύνδεσμο επαλήθευσης που στέλνει η αρχή έκδοσης στον ιδιοκτήτη του domain name μέσω mail. Γι' αυτό το λόγο το πιστοποιητικό εκδίδεται και ενεργοποιείται άμεσα.

Συνιστάται για φυσικά πρόσωπα και μικρές επιχειρήσεις, που χρειάζονται άμεσα ένα πιστοποιητικό SSL και ενδείκνυται για σελίδες που απαιτούν ανταλλαγή πληροφορίας σε κρυπτογραφημένη μορφή π.χ. για σελίδες που υλοποιούν συναλλαγές μικρής κλίμακας.

Με το **Organization Validation** παρέχεται ασφάλεια υψηλότερου επιπέδου αφού, η οντότητα πιστοποιεί τα εταιρικά της στοιχεία και την εταιρική της ταυτότητα, όπως: επωνυμία, πόλη, νομός, χώρα που εδρεύει η επιχείρηση. Σε αυτή την περίπτωση, ενδέχεται να ζητηθούν εταιρικά έγγραφα από την εκδούσα αρχή για την επιβεβαίωση των παραπάνω πληροφοριών.

Συνίσταται για επιχειρήσεις ανεξαρτήτου μεγέθους, οι οποίες έχουν ανάγκη από μεγαλύτερο επίπεδο ασφάλειας, με σκοπό να κερδίσουν τη μέγιστη δυνατή εμπιστοσύνη από τους πελάτες τους και να διατηρήσουν την ανταγωνιστικότητα τους.

Η κατηγορία **Extended Validation** περιλαμβάνει την αυστηρότερη διαδικασία ελέγχων και απαιτεί την κατάθεση εταιρικών εγγράφων προς την εκδούσα αρχή. Η διαδικασία αποτελείται από εφτά επίπεδα και αφορά:

- Την αποκλειστική ιδιοκτησία του domain name
- Την έδρα του οργανισμού
- Τη φυσική και νόμιμη υπόστασή του
- Τη λειτουργία του
- Την επιβεβαίωση ότι ο ίδιος ο οργανισμός αιτήθηκε την έκδοση του SSL καθώς και
- Τη φυσική και νόμιμη ύπαρξη του νομίμου εκπροσώπου.

Συνίσταται για μεγάλους οργανισμούς και επιχειρήσεις, που δραστηριοποιούνται στις online και e-commerce υπηρεσίες.

Τα χαρακτηριστικά μιας ασφαλούς σύνδεσης είναι ορατά από τους περισσότερους χρήστες του διαδικτύου. Σύμφωνα με έρευνα του 2013 από το eMarketer.com, το 60% των χρηστών εγκαταλείπουν το "καλάθι αγοράς" και δεν ολοκληρώνουν μια παραγγελία γιατί δεν εμπιστεύονται την online διαδικασία συναλλαγής. Η χρήση ενός πιστοποιητικού SSL μπορεί να σταματήσει παρόμοια περιστατικά και να κερδίσει την εμπιστοσύνη των χρηστών. Ανάλογα με το πιστοποιητικό που χρησιμοποιείται εμφανίζεται μία ή περισσότερες από τις ακόλουθες **ενδείξεις**: [124]

Λουκέτο: Πρόκειται για ένα διεθνώς αναγνωρισμένο σύμβολο, που εμφανίζεται στη μπάρα διεύθυνσης του περιηγητή και υποδηλώνει ότι η τοποθεσία είναι ασφαλής.

Περιβάλλον https: Το URL μιας ασφαλούς τοποθεσίας ξεκινάει με https, αντί http, όπου το γράμμα **s** υποδηλώνει την ασφάλεια (security).

Πράσινη μπάρα: Το πράσινο χρώμα στη μπάρα διεύθυνσης του περιηγητή υποδηλώνει τη χρήση Extended Validated πιστοποιητικού.

Trust Seals: Εμφανίζονται στην ιστοσελίδα για να υποδείξουν πως οι τοποθεσίες είναι ασφαλείς και ελεγμένες από τους παρόχους.

2.3.5 **Τείχος προστασίας** (Firewall)

Ένα τείχος προστασίας χρησιμοποιείται για να απαγορεύσει σε εξωτερικούς χρήστες να προσπελάσουν ένα δίκτυο. Για να έχει κάποιος χρήστης πρόσβαση στις παρεχόμενες υπηρεσίες του δικτύου θα πρέπει να είναι είτε εσωτερικός χρήστης είτε εξωτερικός που έχει πιστοποιηθεί.

Στόχος ενός τείχους προστασίας είναι να ελέγχει την κίνηση (traffic) των δεδομένων με διάφορες τεχνικές, οι οποίες επιτρέπουν ή απορρίπτουν την είσοδό τους στον κόμβο του δικτύου. Τα δεδομένα αυτά έχουν την μορφή πακέτων (packets) και ελέγχονται με βάση τις πληροφορίες που φέρουν (π.χ. από ποια IP διεύθυνση προήλθαν). Το αν ένα πακέτο θα απορριφτεί ή όχι εξαρτάται από το σχεδιασμό και τις ρυθμίσεις που έχουν γίνει στο τείχος προστασίας. [2]

2.3.6 **Έξυπνη κάρτα (**Smart Card)

Μια έξυπνη κάρτα (smart card) έχει το ίδιο μέγεθος με μια πιστωτική κάρτα αλλά είναι εντελώς διαφορετική. Μια πιστωτική κάρτα είναι απλά ένα κομμάτι πλαστικό και δεν περιλαμβάνει κάτι άλλο στο εσωτερικό της, ενώ μέσα σε μια smart card υπάρχει ένας ενσωματωμένος μικροεπεξεργαστής, κάποια μνήμη και μια διασύνδεση με το εξωτερικό περιβάλλον. Χρησιμοποιούνται περισσότερο στην Ευρώπη παρά στις Ηνωμένες Πολιτείες για τραπεζικές συναλλαγές καθώς επίσης και στο τομέα της υγειονομικής περίθαλψης. Αυτές οι κάρτες μπορούν:

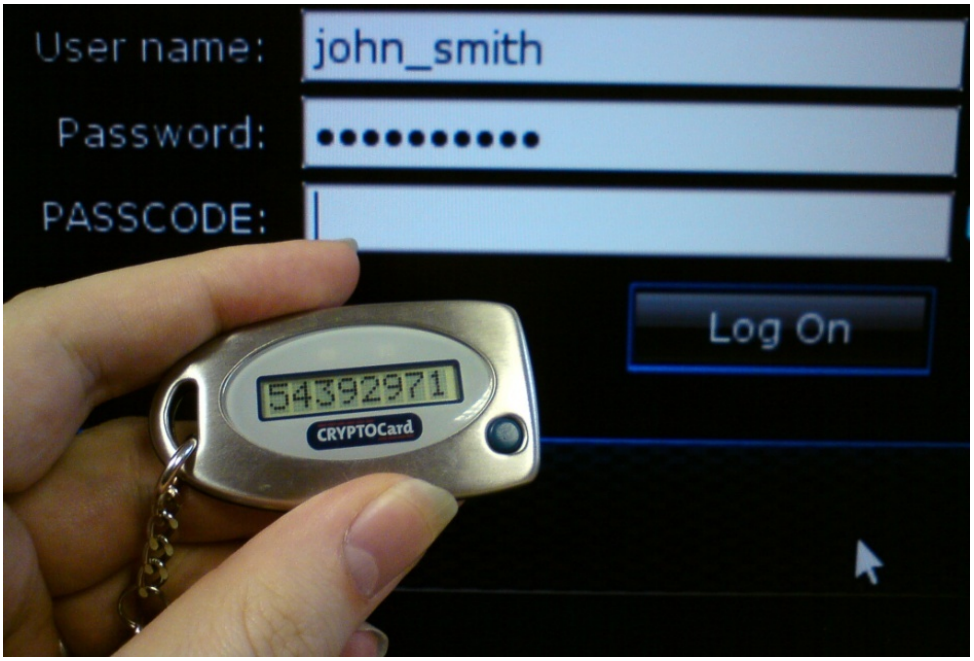
- ✓ να αποθηκεύουν μεγάλες ποσότητες δεδομένων,
- ✓ να περιλαμβάνουν τις δικές τους λειτουργίες (όπως για παράδειγμα λειτουργίες κρυπτογράφησης, αποκρυπτογράφησης και έλεγχου ταυτότητας) και
- ✓ να αλληλεπιδρούν έξυπνα με αντίστοιχες συσκευές ανάγνωσης (readers) σε ένα απομονωμένο περιβάλλον. Υπάρχουν δύο ειδών συσκευές ανάγνωσης, αυτές που διαβάζουν την κάρτα όταν αυτή τοποθετηθεί σε ειδική σχισμή και άλλες που τη διαβάζουν χωρίς επαφή αλλά με την βοήθεια υπέρυθρων ακτινών.

Στις μέρες μας υπάρχει μια μεγάλη γκάμα από smart cards, οι οποίες μεταξύ τους διαφέρουν ως προς την απόδοση και την ικανότητα του επεξεργαστή, το μέγεθος της μνήμης καθώς επίσης και ως προς τη ταχύτητα διασύνδεσης με το εξωτερικό περιβάλλον. [125] [126] [122]

2.3.7 **Μάρκες ασφαλείας (**Security Tokens)

Μια μάρκα ασφαλείας (security token) είναι μια πολύ μικρή συσκευή που αναπαράγει ανά τακτά χρονικά διαστήματα κάποια δεδομένα. Τα δεδομένα αυτά συνήθως είναι αριθμοί και χρησιμοποιούνται ως ένας επιπρόσθετος κωδικός πρόσβασης ο οποίος εισάγεται μαζί με τα

υπόλοιπα διαπιστευτήρια του χρήστη κατά την διαδικασία σύνδεσης (login). Η συσκευή αυτή είναι χαμηλή σε κόστος και κάθε χρήστης έχει τη δική του.



Εικόνα 2.15 - Security Token [156]

Υπάρχουν αρκετές παραλλαγές του Security Token. Η πιο γνωστή είναι το Software Token, το οποίο είναι μια εφαρμογή που λειτουργεί σε smartphones. Αναπαράγει και αυτό ανά τακτά χρονικά διαστήματα έναν κωδικό πρόσβασης και παρέχεται δωρεάν στους χρήστες. [127] [128]

2.3.8 **Πολλαπλά κριτήρια πιστοποίησης**

Η χρήση πολλαπλών κριτηρίων πιστοποίησης (Multi-Factor Authentication) είναι μια μέθοδος ασφάλειας που χρησιμοποιείται για την πιστοποίηση της ταυτότητας ενός χρήστη. Σύμφωνα με αυτή την μέθοδο, υπάρχουν τρεις κύριες κατηγορίες που έχουν να κάνουν με **αυτό που είναι**, με **αυτό που γνωρίζει** και **αυτό που έχει** ένας χρήστης.

Η πρώτη κατηγορία είναι η πιο ασφαλής. Περιλαμβάνει την αναγνώριση φωνής, την αμφιβληστροειδή σάρωση, τη λήψη δακτυλικών αποτυπωμάτων και άλλες βιομετρικές τεχνικές.

Η κατηγορία που έχει να κάνει με **αυτό που γνωρίζει** ένας χρήστης περιλαμβάνει συνήθως τη χρήση **κωδικού πρόσβασης** ή την απάντηση σε μια ερώτηση ασφαλείας. Σε αυτή την περίπτωση ένας χρήστης είναι λιγότερο ασφαλής αφού μπορεί να χάσει ή να ξεχάσει αυτό που γνωρίζει.

Αυτό που έχει ένας χρήστης μπορεί να είναι μια **smart card**,ένα **security token** ή κάποιο άλλο υλικό αγαθό.

Για να εξασφαλιστεί η μέγιστη δυνατή ασφάλεια ενός συστήματος, θα πρέπει να συνδυαστούν και οι τρεις κατηγορίες. Ωστόσο, οι περισσότερες διαδικτυακές εφαρμογές συνδυάζουν τις δύο τελευταίες. Κατά συνέπεια, η μέθοδος αυτή είναι επίσης γνωστή ως μέθοδος πιστοποίησης δύο κριτηρίων (Two-Factor Authentication). Ένα κλασικό παράδειγμα αυτής της μεθόδου μπορεί να συναντήσει κάποιος σε ιστοσελίδες τραπεζών και τζόγου. Σε αυτές τις ιστοσελίδες ένας χρήστης χρησιμοποιεί ένα **κωδικό πρόσβασης** και ένα **security token**. [89] [129]

2.3.9 Εικονικό δίκτυο περιήγησης

Ένα εικονικό δίκτυο περιήγησης (Virtual Private Network ή VPN) δημιουργείται πάνω από μια υπάρχουσα φυσική δικτυακή υποδομή και συνδυάζει τα χαρακτηριστικά ενός τοπικού δικτύου (LAN) μαζί με μηχανισμούς που προσφέρουν κρυπτογραφημένη σύνδεση (όπως για παράδειγμα ένα πρωτόκολλο SSL) προκειμένου να επικοινωνούν με ασφάλεια υπολογιστές από ολόκληρο τον κόσμο. Αυτό συνεπάγεται ότι από όπου και αν συνδέεται ένας υπολογιστής, η σύνδεσή του αποκτάει τα χαρακτηριστικά της βάσης δεδομένων του VPN, όπως π.χ. την εξωτερική IP διεύθυνση.

Η χρήση ενός VPN είναι εύκολη και μεταξύ άλλων ευνοεί την ασφαλή πρόσβαση στο οικιακό και εργασιακό δίκτυο ενός χρήστη από απόσταση, την ανώνυμη περιήγηση στο διαδίκτυο καθώς επίσης και την πρόσβαση σε ιστοσελίδες οι οποίες επιτρέπουν την είσοδο μόνο σε χρήστες από μια συγκεκριμένη χώρα. [130] [131] [132]

Μερικά από τα πιο γνωστά και οικονομικά VPN είναι τα εξής:

- TunnelBear
- proXPN
- VPNPOP
- VPN Reactor
- CactusVPN

3 ΚΕΦΑΛΑΙΟ - Η Εφαρμογή

3.1 Γενική περιγραφή

Η εφαρμογή αυτή έχει ως αντικείμενο τη συλλογή πληροφοριών σχετικά με τις αδυναμίες, τις επιθέσεις και τα μέτρα προστασίας που οφείλει να γνωρίζει ένας προγραμματιστής διαδικτυακών εφαρμογών. Δεν βασίζεται σε κάποιο σύστημα διαχείρισης περιεχομένου (CMS) ή Framework. Ωστόσο, παρέχει την ευελιξία που παρέχουν και τα υπόλοιπα CMS / Framework επιτρέποντας την προσθήκη επιπλέον πρόσθετων (extensions): component, module και template. Όπως και στα υπόλοιπα συστήματα διαχείρισης περιεχομένου, έτσι και στην παρούσα εφαρμογή:

Ένα **component** αποτελεί μια μικρή ενότητα λειτουργιών μέσα στην ίδια την εφαρμογή.

Ένα **module** εμφανίζεται δίπλα από ένα ή περισσότερα component και δίνει επιπλέον λειτουργικότητα στην εφαρμογή.

Ένα **template** δίνει την δυνατότητα να αλλάξει κάποιος τη δόμηση της ιστοσελίδας χωρίς να υποστούν καμία αλλαγή τα δεδομένα και οι λειτουργικότητά της.

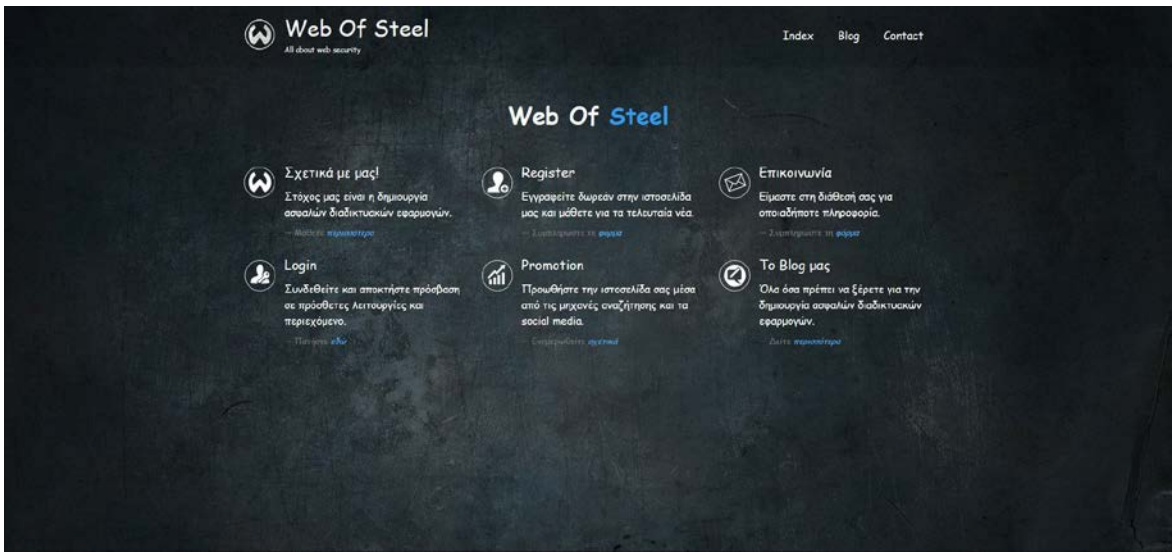
3.2 Λειτουργικά χαρακτηριστικά

Η εφαρμογή είναι σχεδιασμένη έτσι ώστε κάθε χρήστης να ανήκει σε μια μόνο κατηγορία. Σε κάθε μια από αυτές τις κατηγορίες αντιστοιχεί ένα επίπεδο πρόσβασης (access level) το οποίο καθορίζει ποιες λειτουργίες μπορεί να χρησιμοποιήσει ένας χρήστης και ποιες όχι. Στον πίνακα που ακολουθεί αναφέρονται συνοπτικά τα χαρακτηριστικά κάθε κατηγορίας.

Επίπεδο πρόσβασης	Κατηγορία	Περιγραφή
1	Επισκέπτης	Η κατηγορία με τα ελάχιστα δικαιώματα
3	Εγγεγραμμένος χρήστης	Η κατηγορία που έχει πρόσβαση σε περιεχόμενο που ένας απλός επισκέπτης δεν έχει.
9	Διαχειριστής	Ο διαχειριστής έχει όλα τα δικαιώματα που έχει ένας εγγεγραμμένος χρήστης. Επιπροσθέτως, μπορεί να κάνει εισαγωγή, τροποποίηση και διαγραφή του περιεχομένου της εφαρμογής.

3.2.1 Index Component

Η πρώτη σελίδα εμφανίζει συνοπτικά το περιεχόμενο και τις λειτουργίες της εφαρμογής. Σε όλες τις σελίδες εμφανίζεται πάνω αριστερά το όνομα και το λογότυπο ενώ πάνω δεξιά το βασικό μενού.



Εικόνα 3.1 - Index Component

3.2.2 Blog Component

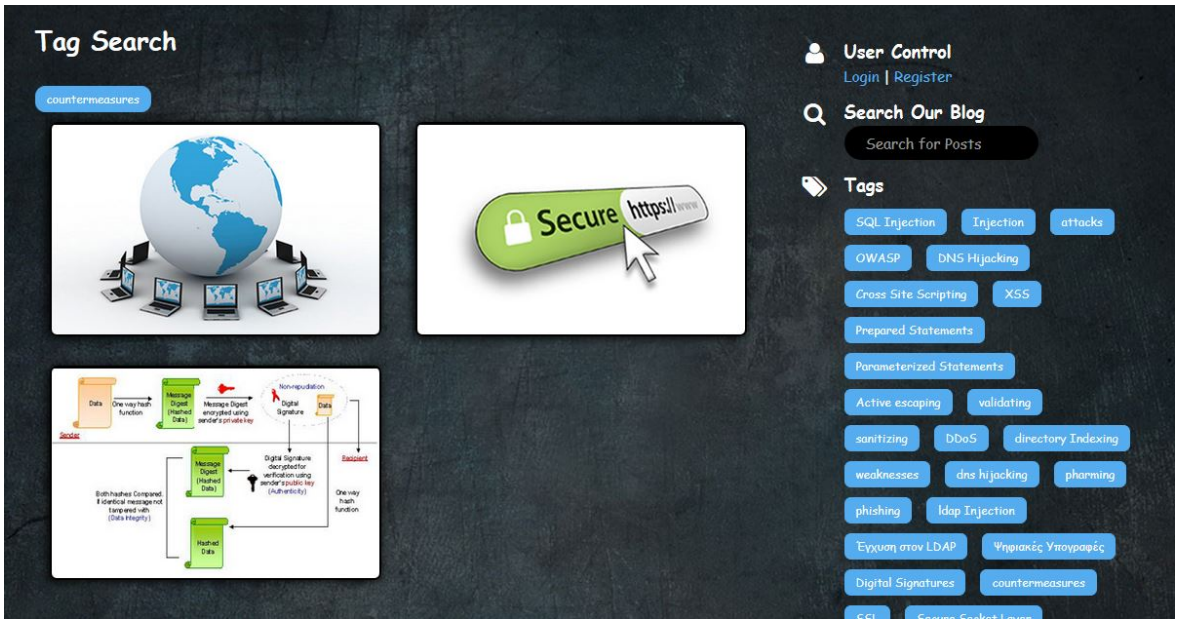
Κάνοντας κλικ στην επιλογή "Blog", ο χρήστης μπορεί να μεταφερθεί στην αντίστοιχη σελίδα (component) και να επιλέξει να διαβάσει κάποια ανάρτηση (Blog Post).

Εικόνα 3.2 - Blog Component

Η εφαρμογή είναι σχεδιασμένη με τέτοιο τρόπο που να εμφανίζει τις αναρτήσεις με βάση την ημερομηνία που αποθηκεύτηκαν στη Βάση Δεδομένων (ξεκινώντας πάντα από την πιο πρόσφατη). Στα δεξιά εμφανίζονται τα εξής **modules**:

- Ο έλεγχος χρήστη (User control): που επιτρέπει την εγγραφή (Register) / σύνδεση (Log In) σε έναν χρήστη.
- Τη μπάρα αναζήτησης (search bar) καθώς επίσης και
- Τη δυνατότητα αναζήτησης με βάση τις λέξεις κλειδιά (Tags)

Εικόνα 3.3 - Search Bar Module



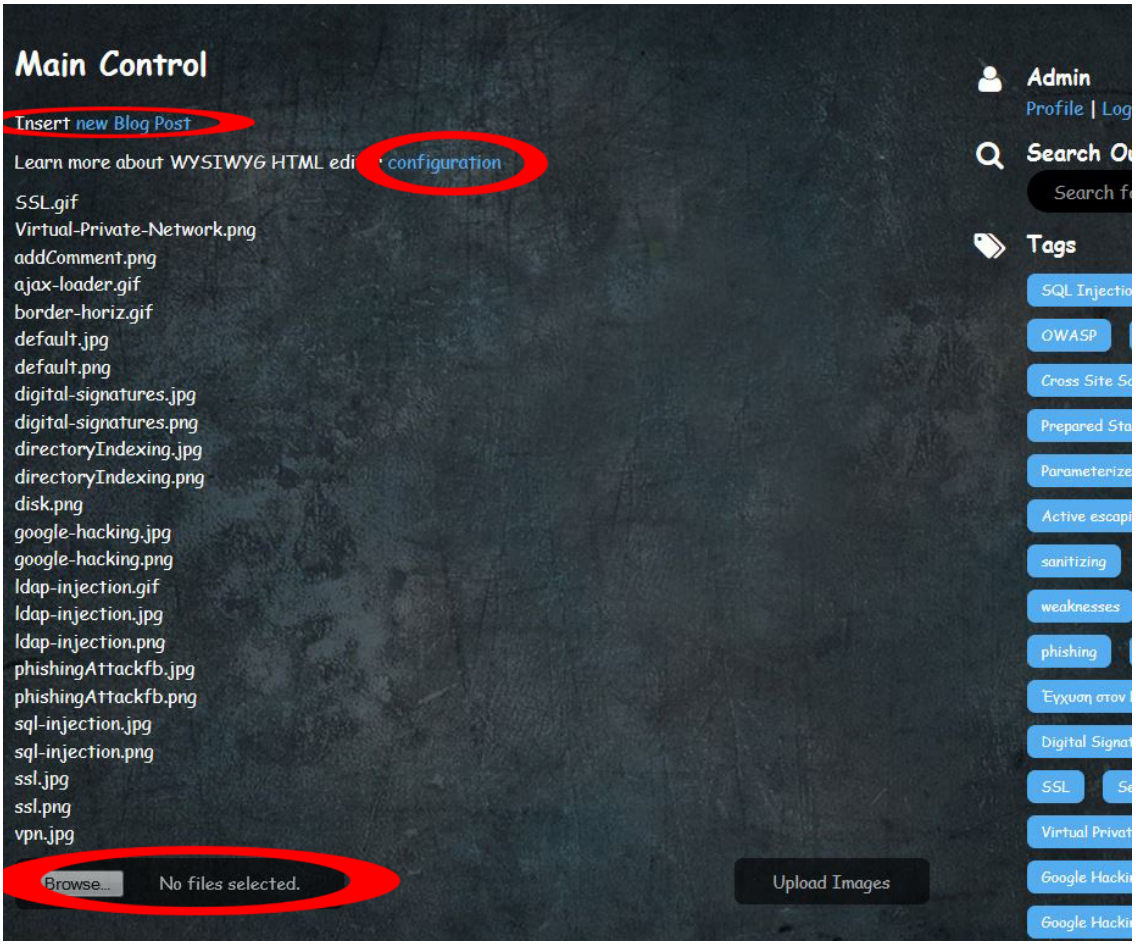
Εικόνα 3.4 - Search By Tag Module

Σε περίπτωση που ένας χρήστης ανήκει στην κατηγορία του Διαχειριστή (9ο επίπεδο πρόσβασης) και είναι ήδη συνδεδεμένος (Logged In) θα εμφανιστεί στη σελίδα των αναρτήσεων (Blog Posts) μια επιπρόσθετη επιλογή με όνομα "**control panel**".

Εικόνα 3.5 - Option: Control Panel

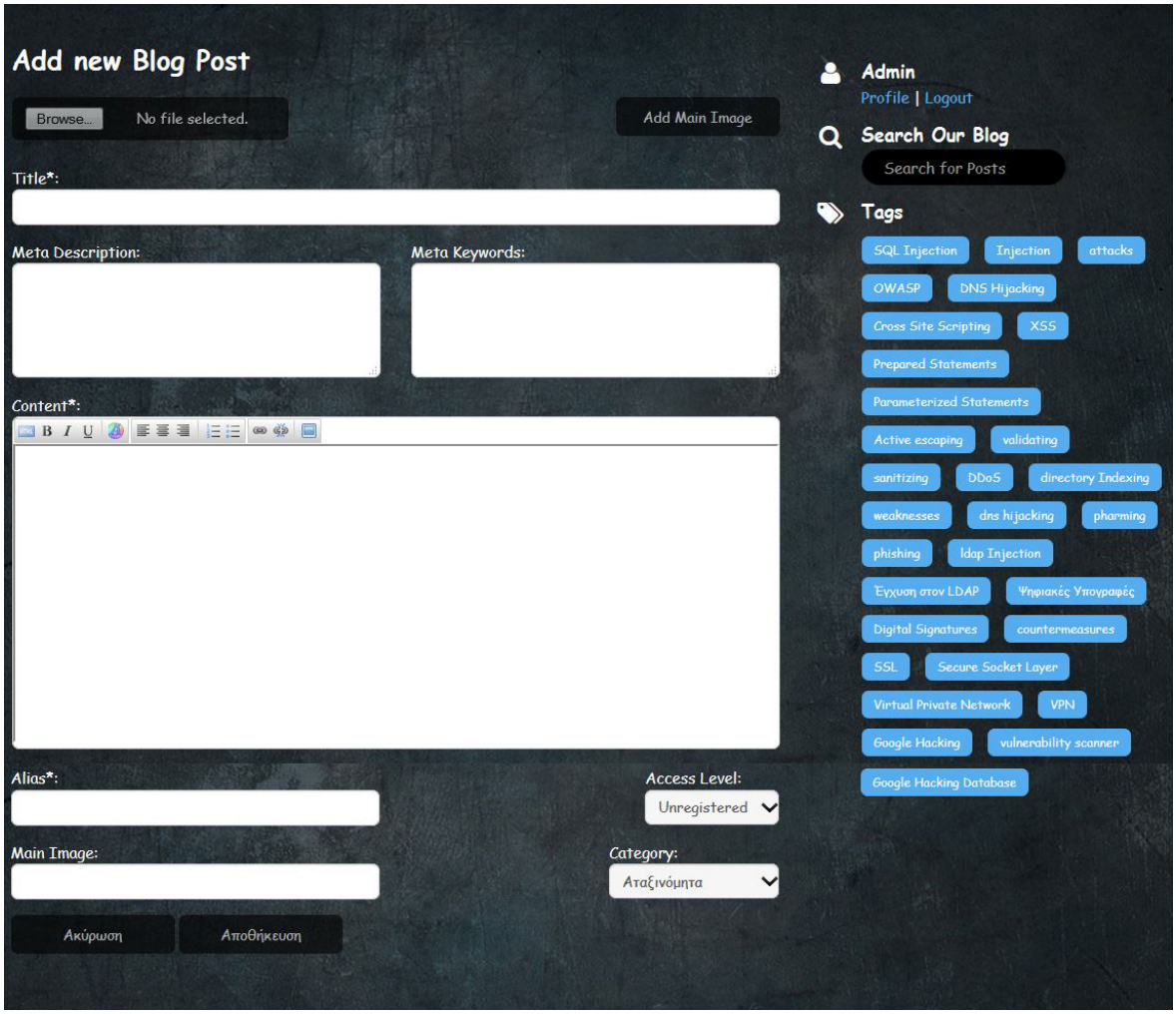
Κάνοντας κλικ στο control panel εμφανίζονται τα εξής:

- Η δυνατότητα εισαγωγής νέας ανάρτησης (**Insert new Blog Post**)
- Ένας σύνδεσμος που παραπέμπει στην ιστοσελίδα jhtmlarea.codeplex.com. Ο σύνδεσμος αυτός παρέχει χρήσιμες πληροφορίες για τον WYSIWYG επεξεργαστή ο οποίος χρησιμοποιείται στις λειτουργίες εισαγωγής (insert new blog post) και επεξεργασίας (edit blog post) των αναρτήσεων (όπως θα δούμε και στη συνέχεια).
- Μια λίστα με τα ονόματα των εικόνων που έχουν αποθηκευτεί στον εξυπηρετητή και εμφανίζονται σε παλαιότερες αναρτήσεις καθώς επίσης και
- Η δυνατότητα αποθήκευσης πολλαπλών εικόνων στον αντίστοιχο φάκελο που βρίσκεται στον εξυπηρετητή (**Upload Images**).



Εικόνα 3.6 - Control Panel Functionalities

Πατώντας στο σύνδεσμο "**Insert new Blog Post**" ο διαχειριστής μεταφέρεται στην αντίστοιχη σελίδα.

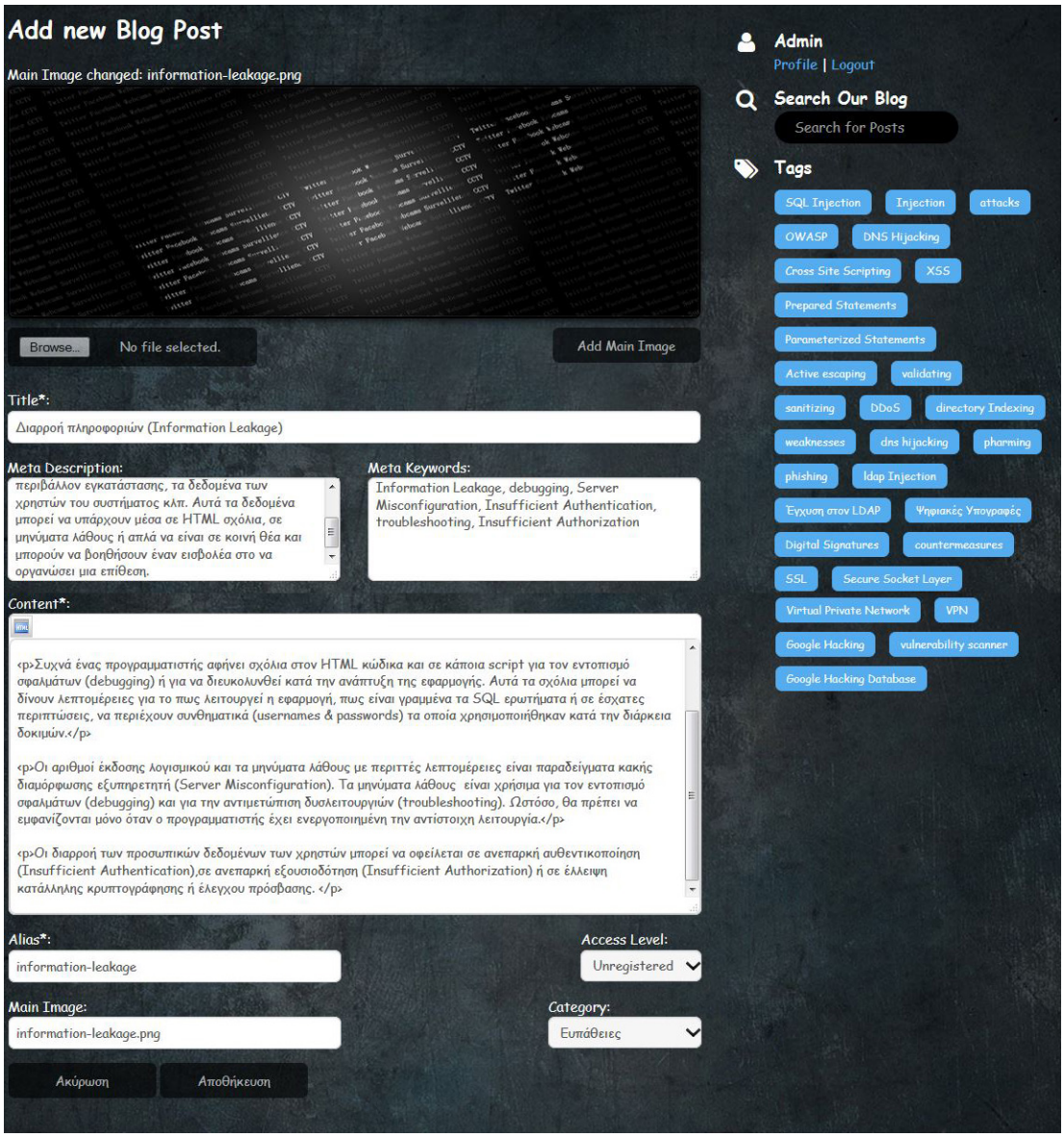


Εικόνα 3.7 - Add New Blog Post

Για κάθε νέα ανάρτηση συμπληρώνονται τα εξής πεδία:

- Το υποχρεωτικό πεδίο **Title** περιλαμβάνει τον τίτλο της ανάρτησης.
- Τα πεδία **Meta Description** και **Meta Keywords** περιλαμβάνουν τα δεδομένα που θα συμπεριληφθούν στα αντίστοιχα html meta elements. Παρόλο που τα πεδία αυτά δεν είναι υποχρεωτικά θα πρέπει να συμπληρώνονται. Εισάγοντας τα σωστά δεδομένα / λέξεις κλειδιά κάθε μια από αυτές τις αναρτήσεις θα εμφανίζεται τόσο στα αποτελέσματα των μηχανών αναζήτησης (π.χ. Google.com) όσο και στα αποτελέσματα των εσωτερικών μηχανισμών αναζήτησης (όπως συμβαίνει με τους μηχανισμούς / modules: search Bar και search Tag που αναφέρθηκαν παραπάνω).
- Στο υποχρεωτικό πεδίο **Content** συμπεριλαμβάνεται όλο το υλικό της ανάρτησης (κείμενο, εικόνες κλπ). Το πεδίο αυτό είναι ειδικά διαμορφωμένο από τον WYSIWYG επεξεργαστή ο οποίος δίνει την δυνατότητα επεξεργασίας του περιεχομένου σε χρήστες που δεν γνωρίζουν τους κανόνες της HTML.
- Στο υποχρεωτικό πεδίο **alias** συμπληρώνεται η ακριβής URL διεύθυνση της ανάρτησης (π.χ. information-leakage).

- Το πεδίο **Main Image** μπορεί να συμπληρωθεί είτε χειροκίνητα, εισάγοντας κάποιο από τα ονόματα των εικόνων που βρίσκονται στην λίστα εικόνων της σελίδας (**control panel**), είτε από τον μηχανισμό εισαγωγής εικόνων (**Add Main Image**) που βρίσκεται πάνω από το πεδίο **Title**. Αν το πεδίο δεν συμπληρωθεί θα εμφανιστεί μια προεπιλεγμένη εικόνα.
- Στο πεδίο **access level** ο χρήστης μπορεί να επιλέξει αν η ανάρτηση θα είναι ορατή από όλες τις κατηγορίες χρηστών ή όχι.
- Το πεδίο **category** καθορίζει την κατηγορία στην οποία ανήκει το άρθρο.

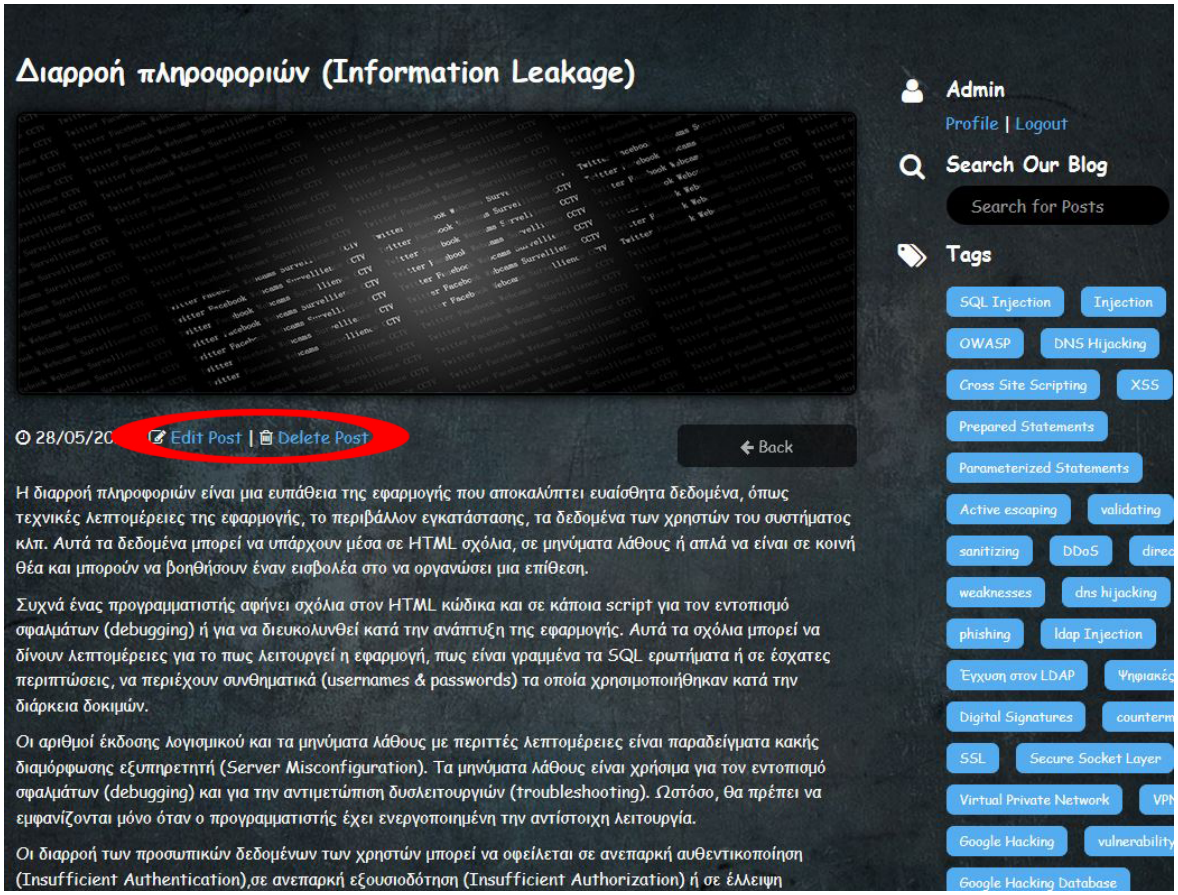


Εικόνα 3.8 - Add New Blog Post Example

Το αποτέλεσμα της παραπάνω εισαγωγής εμφανίζεται αυτόματα στη σελίδα αναρτήσεων.

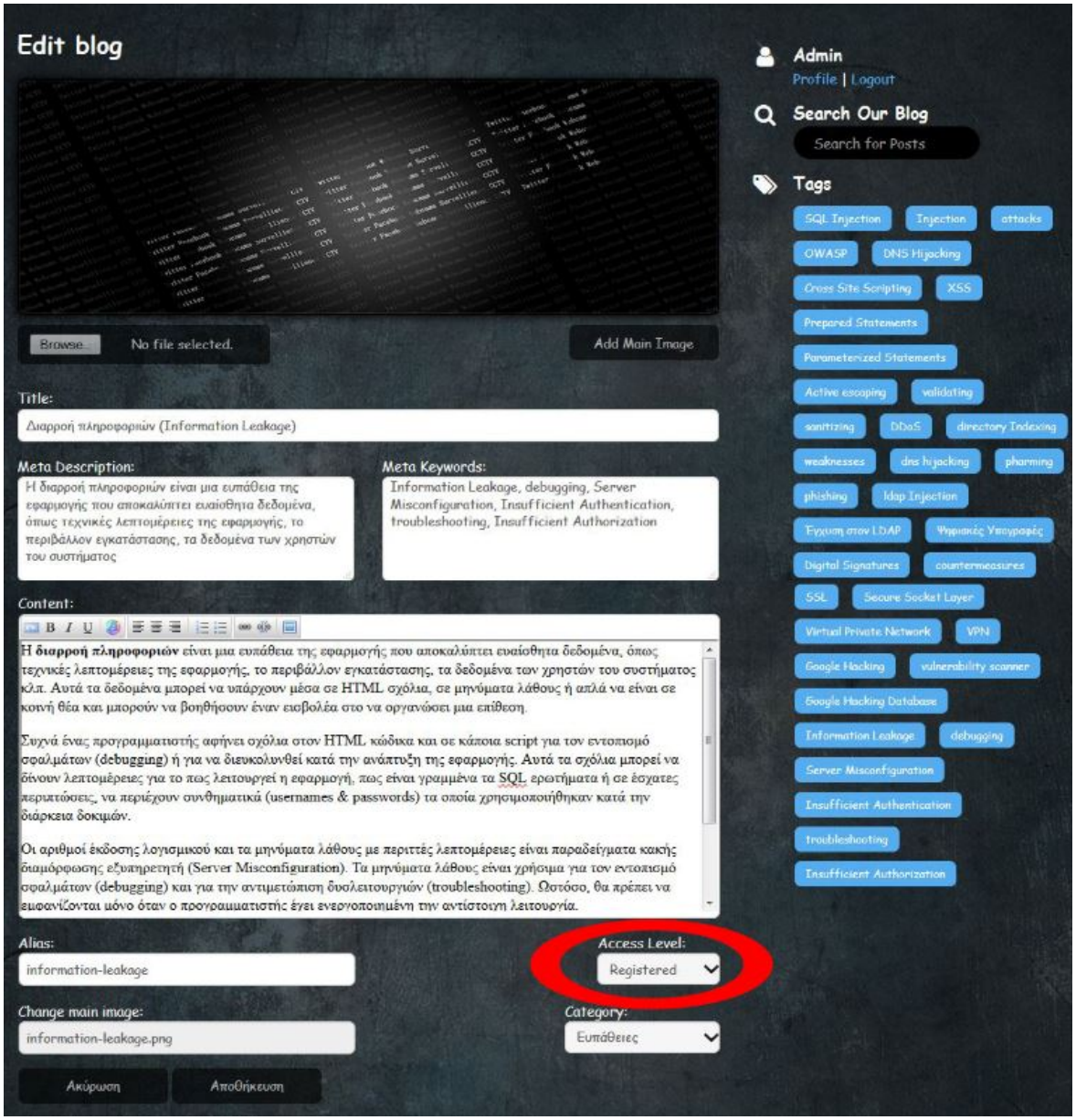
Εικόνα 3.9 - Blog Component Main Page

Κάνοντας κλικ σε μια εικόνα, ο χρήστης μπορεί να μεταβεί στην αντίστοιχη ανάρτηση. Αν ο χρήστης είναι συνδεδεμένος ως διαχειριστής τότε μπορεί να επεξεργαστεί ή να διαγράψει την αντίστοιχη ανάρτηση.



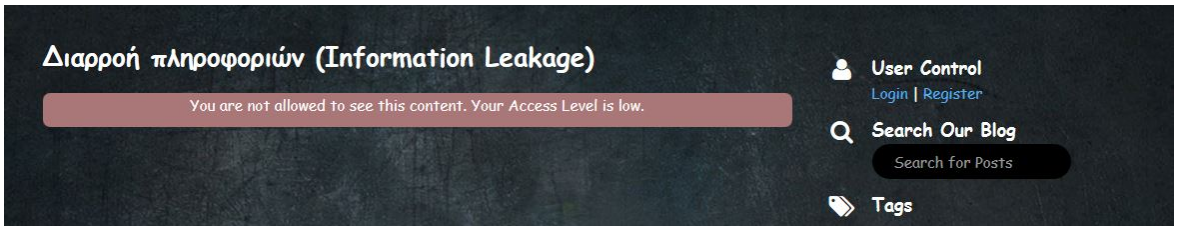
Εικόνα 3.10 - Edit / Delete Current Blog Post

Επιλέγοντας **Edit Post** ο χρήστης μπορεί να μεταβεί στην αντίστοιχη σελίδα και να υποβάλει οποιαδήποτε αλλαγή θέλει. Για παράδειγμα, μπορεί να επιλέξει να εμφανίζεται η συγκεκριμένη ανάρτηση μόνο στους χρήστες που ανήκουν στην κατηγορία "εγγεγραμμένα μέλη"



Εικόνα 3.11 - Edit Blog Post

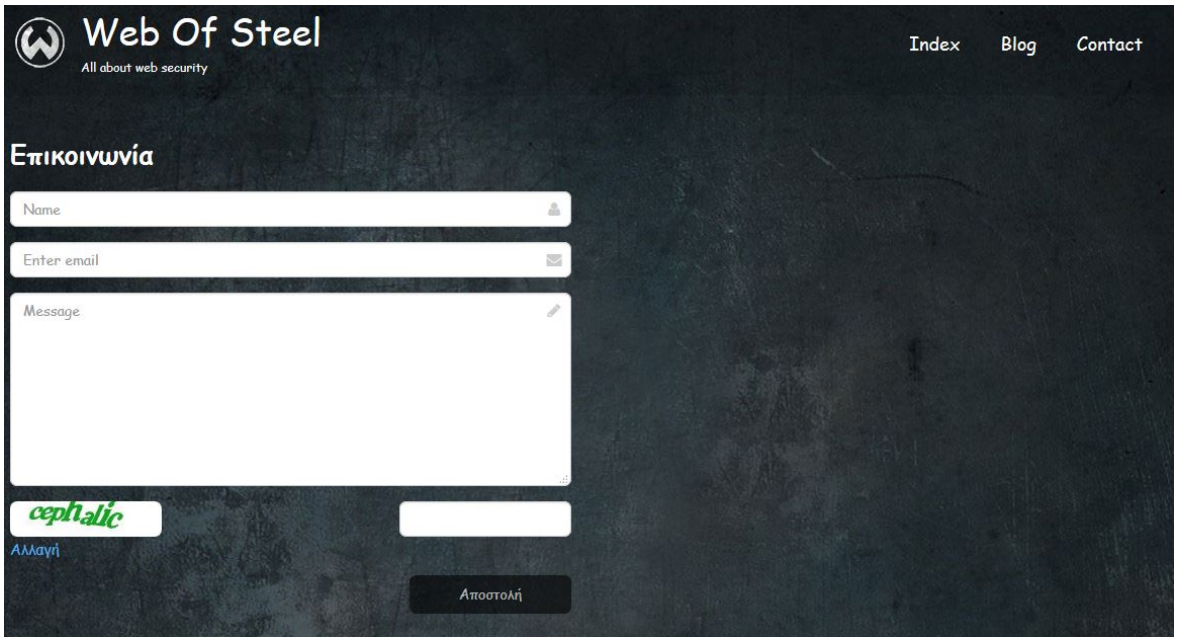
Σε περίπτωση που ένας χρήστης δεν ανήκει στην κατηγορία "εγγεγραμμένα μέλη" και προσπαθήσει να μεταβεί στην αντίστοιχη σελίδα είτε πληκτρολογώντας το ακριβές URL στην μπάρα διευθύνσεων του περιηγητή του (https://www.webofsteel.com/information-leakage) είτε με οποιοδήποτε άλλο τρόπο θα του απαγορευτεί η πρόσβαση.



Εικόνα 3.12 - Low Access Level

3.2.3 Contact Component

Πατώντας "**Contact**" από το κύριο μενού οποιοσδήποτε χρήστης μπορεί να μεταβεί στη φόρμα επικοινωνίας και να στείλει κάποιο μήνυμα.

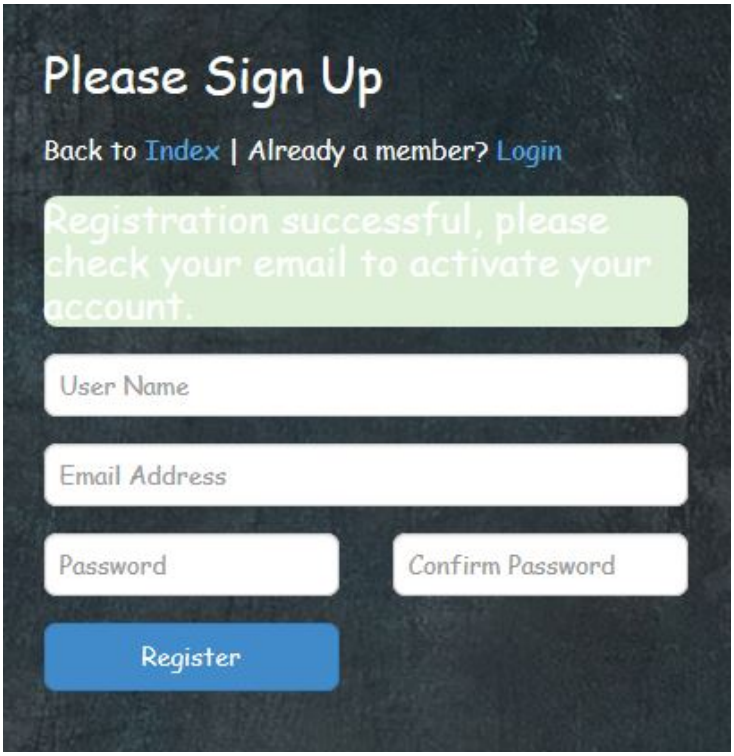


Εικόνα 3.13 - Contact Component

3.2.4 User Component

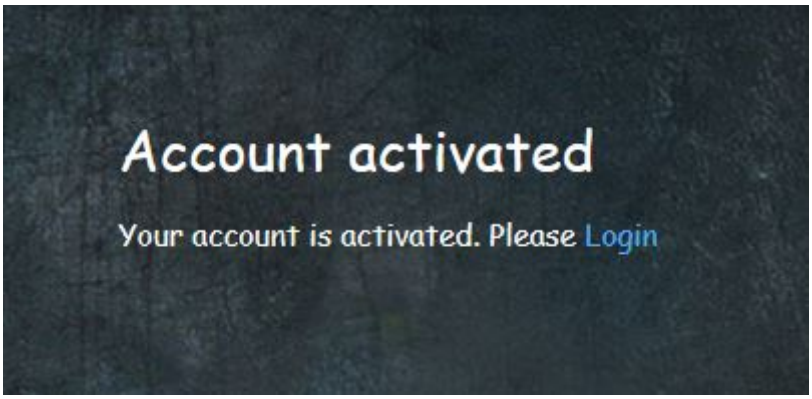
Ο μηχανισμός User περιλαμβάνει όλες τις λειτουργίες που αφορούν τους χρήστες. Κάθε μια από αυτές τις λειτουργίες ακολουθεί κάποιους κανόνες ασφαλείας οι οποίοι αναφέρονται αναλυτικά στην παράγραφο (3.5 Μηχανισμοί ασφαλείας).

Επιλέγοντας **Register** ένας χρήστης μπορεί να εισάγει το όνομα του, τον κωδικό πρόσβασης και το email του. Αφού πατήσει το κουμπί "Register" και δεν υπάρξει οποιαδήποτε επιπλοκή, θα εμφανιστεί το αντίστοιχο μήνυμα.



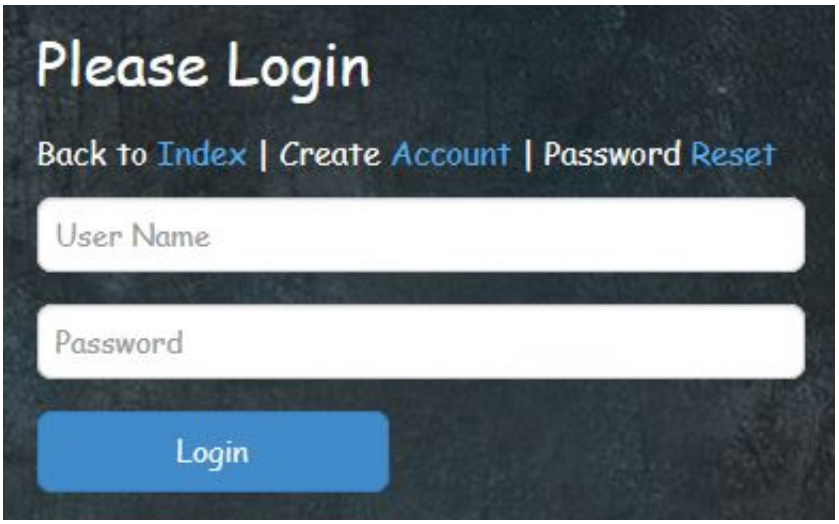
Εικόνα 3.14 - Sign Up Form

Στη συνέχεια, θα αποσταλεί ένα μήνυμα επιβεβαίωσης στο email του χρήστη με έναν σύνδεσμο τον οποίο θα πρέπει να επισκεφτεί προκειμένου να επιβεβαιώσει ότι το email του είναι ενεργό.



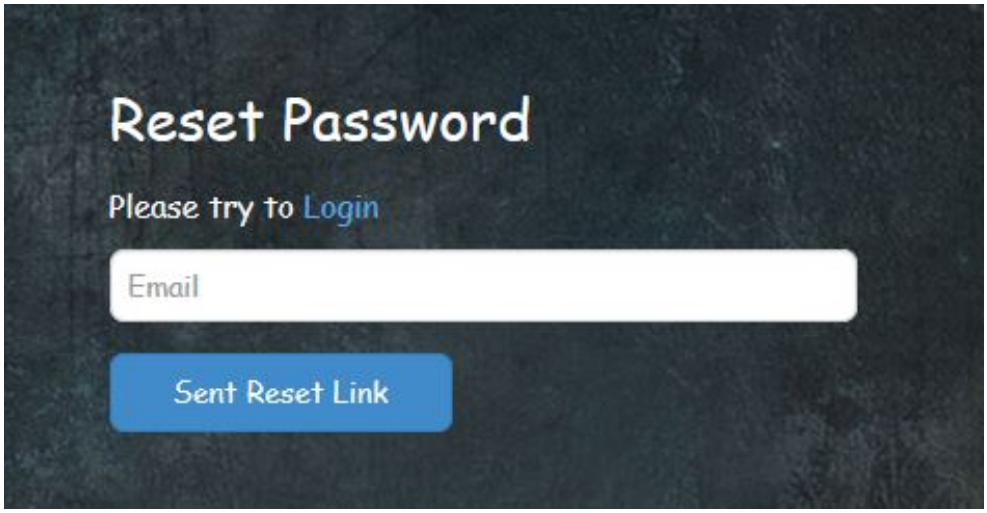
Εικόνα 3.15 Account Activated

Όταν ένας χρήστης ενεργοποιήσει το λογαριασμό του, μπορεί να συνδεθεί μέσα από την αντίστοιχη φόρμα.



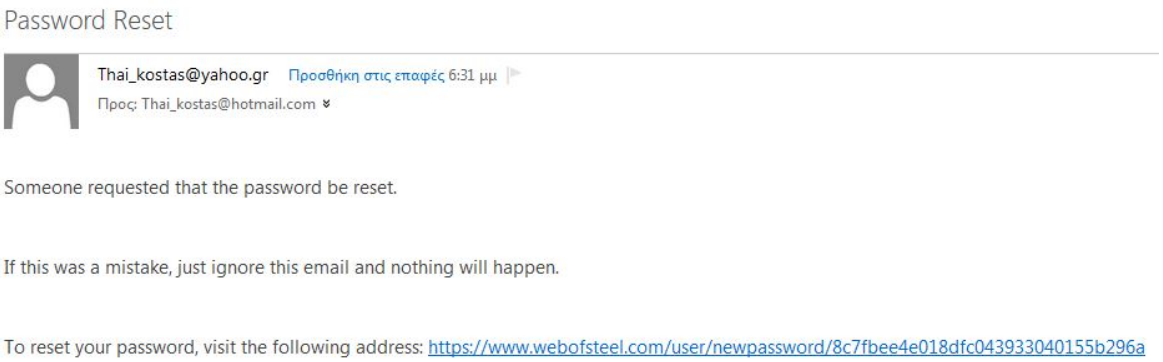
Εικόνα 3.16 Log In Form

Μια ακόμα λειτουργία που μπορεί να χρησιμοποιήσει ένας εγγεγραμμένος χρήστης είναι η ανάκτηση του κωδικού πρόσβασης. Πατώντας πάνω στο σύνδεσμο **Password Reset** εμφανίζεται η αντίστοιχη φόρμα.



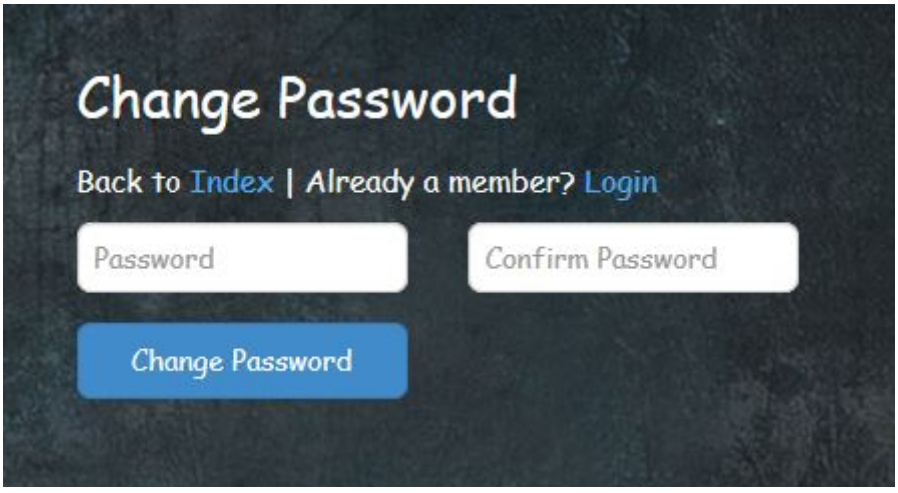
Εικόνα 3.17 - Reset Pasword

Αφού εισάγει ένας χρήστης το email του θα του αποσταλεί σχετικό μήνυμα με έναν σύνδεσμο.



Εικόνα 3.18 - Email for New Password

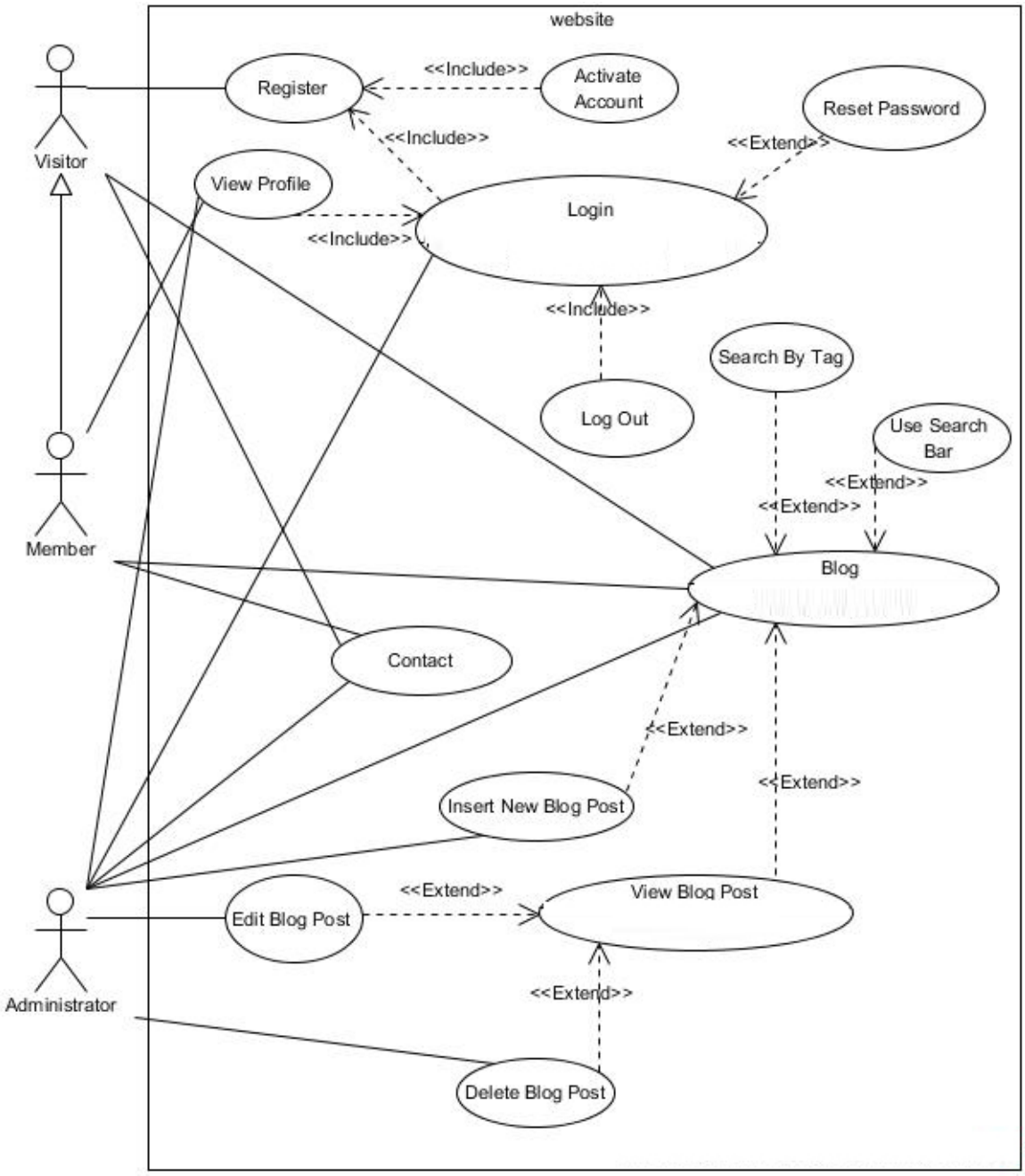
Κάνοντας κλικ σε αυτό το σύνδεσμο θα μεταφερθεί σε μια φόρμα όπου θα του ζητηθεί να συμπληρώσει έναν νέο κωδικό πρόσβασης.



Εικόνα 3.19 - Change Password

3.3 **Διαγράμματα** UML

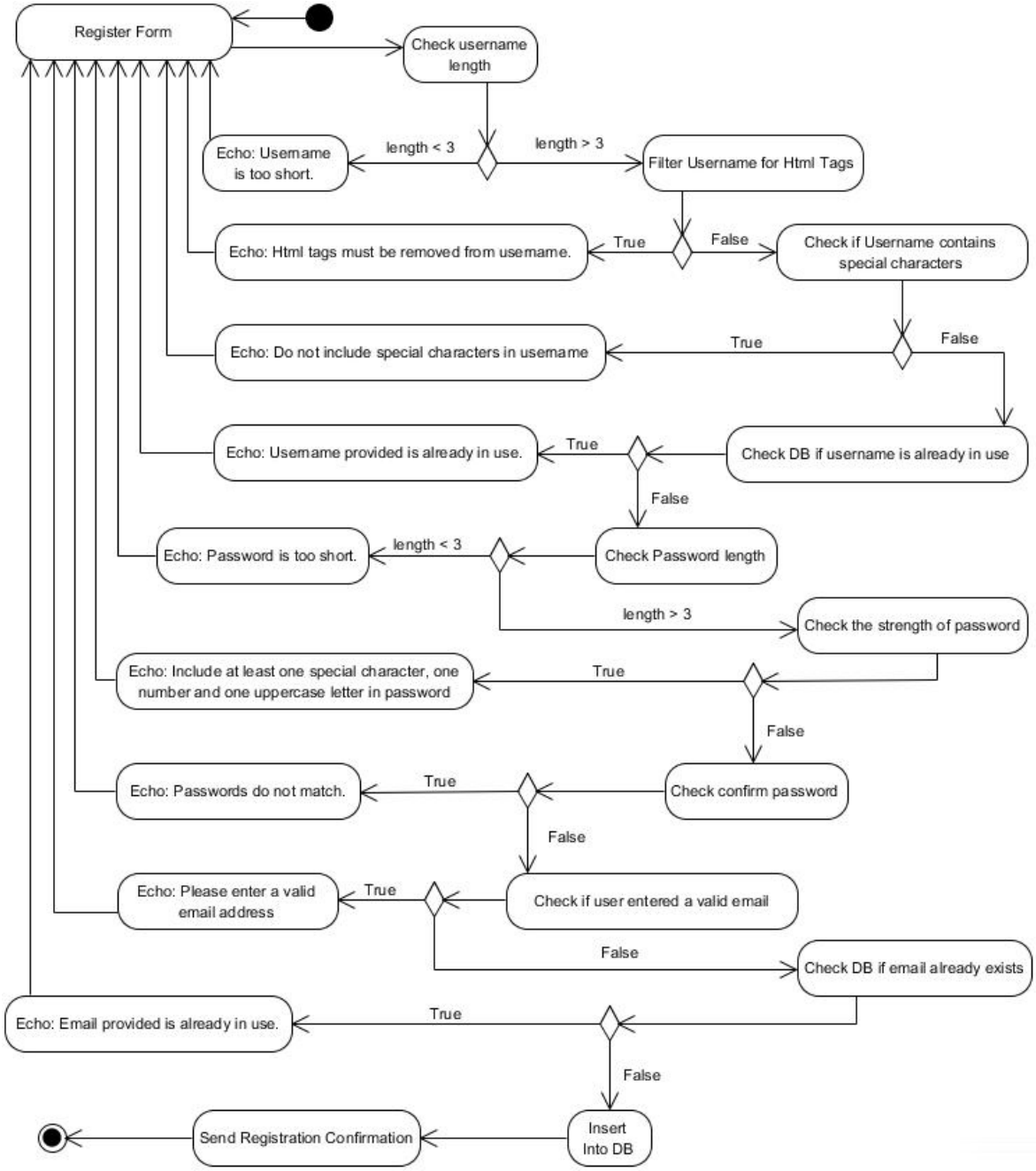
3.3.1 **Διάγραμμα περίπτωσης χρήσης (**Use Case)



Εικόνα 3.20 Use Case Diagram

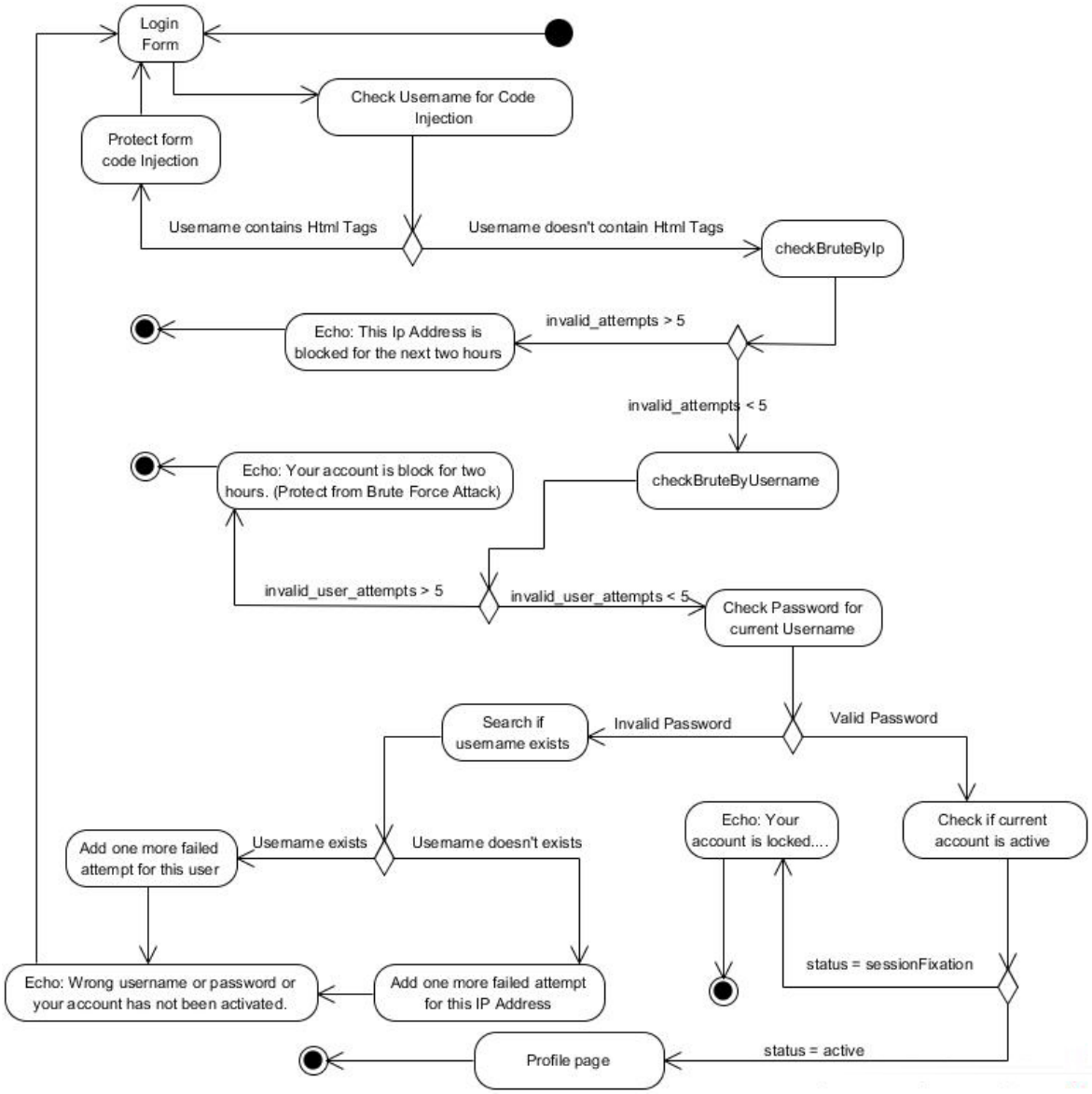
3.3.2 **Διαγράμματα δραστηριοτήτων**

3.3.2.1 **Διάγραμμα δραστηριότητας "**Register"



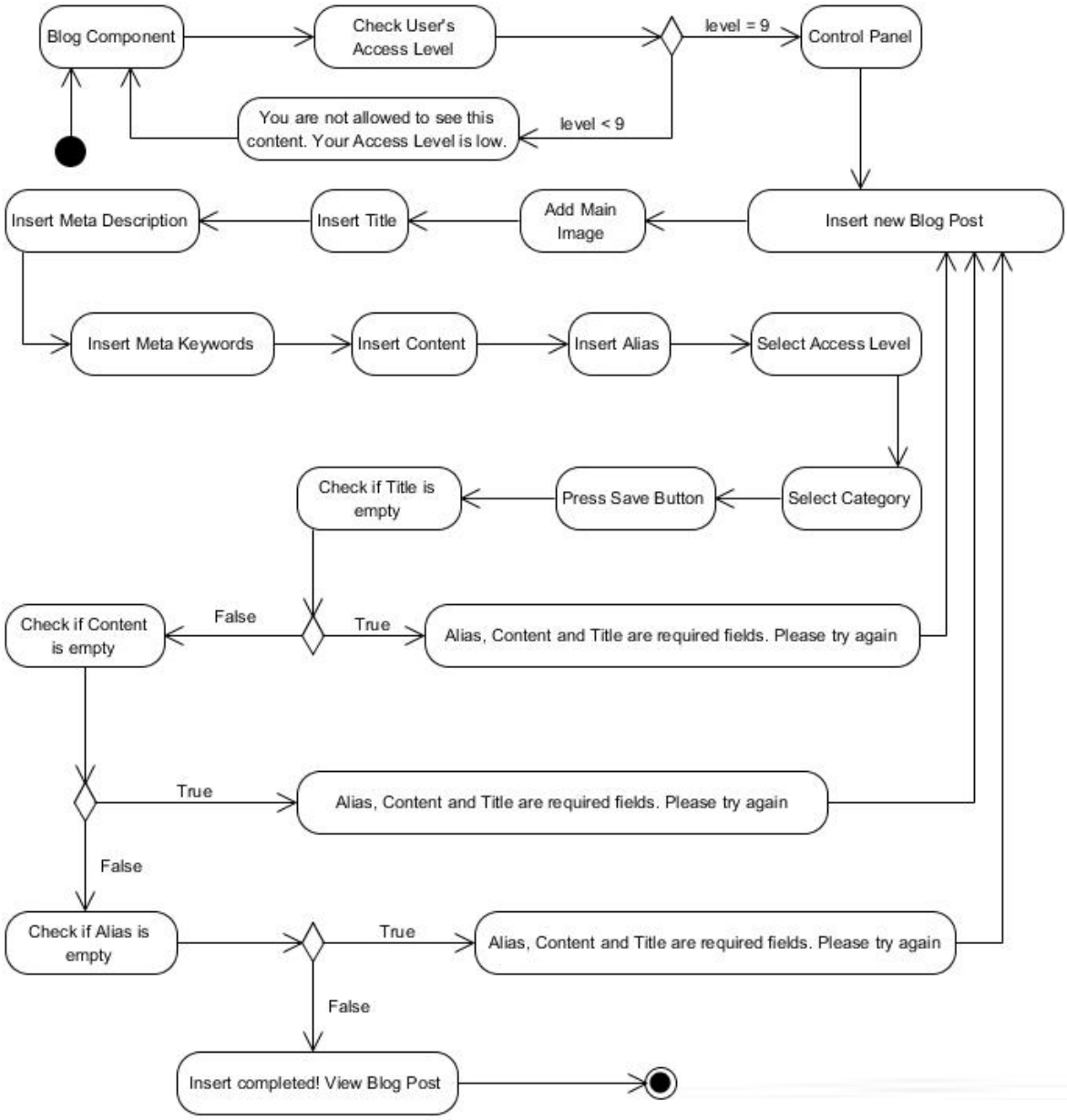
Εικόνα 3.21 Activity diagram for Register

3.3.2.2 **Διάγραμμα δραστηριότητας** "Login"



Εικόνα 3.22 Activity Diagram for Login

3.3.2.3 **Διάγραμμα δραστηριότητας** "Insert new Blog Post"



Εικόνα 3.23 Activity Diagram Insert new Blog Post

3.4 Τεχνολογικά χαρακτηριστικά

Σε αυτή την παράγραφο αναφέρονται όλες οι τεχνολογίες που έχουν χρησιμοποιηθεί για την υλοποίηση αυτής της εφαρμογής. Πρόκειται για μια εφαρμογή που στο μεγαλύτερο μέρος της δεν βασίζεται ούτε σε κάποιο γνωστό framework, όπως π.χ. στο Zend, στο Symfony, στο prado κλπ., ούτε σε κάποιο έτοιμο σύστημα διαχείρισης περιεχομένου (Content Management System - CMS), όπως π.χ. στο Joomla,στο Drupal κλπ.. Ουσιαστικά, αποτελεί ένα αυτοσχέδιο framework που μπορεί να παρέχει τις ίδιες δυνατότητες που παρέχει ένα CMS, ευνοώντας παράλληλα τη δημιουργία επιπρόσθετων components, modules και templates.

Οι τεχνολογίες που έχουν χρησιμοποιηθεί εντάσσονται στην κατηγορία λογισμικού ανοιχτού κώδικα (OpenSource Software) και αναφέρονται συνοπτικά παρακάτω. [133]

3.4.1 Εξυπηρετητής Apache

Ο **Apache** είναι ένας δημοφιλής HTTP εξυπηρετητής που μπορεί να εγκατασταθεί σε διάφορα λειτουργικά συστήματα όπως για παράδειγμα σε Windows και Linux. Χρησιμοποιείται από πάρα πολλούς πάροχους φιλοξενίας ιστοσελίδων (web hosting) και είναι παραμετροποιήσιμος σε μεγάλο βαθμό. Για λόγους ασφαλείας, η εγκατάσταση και παραμετροποίηση του πρέπει να γίνεται από έμπειρους χρήστες. [2]

3.4.2 Γλώσσα PHP

Η **PHP** είναι μια γλώσσα προγραμματισμού που χρησιμοποιείται στην κατασκευή scripts για δυναμικές ιστοσελίδες. Λειτουργεί από την πλευρά του εξυπηρετητή και μπορεί να ενσωματωθεί εύκολα σε κώδικα HTML.Μερικά από τα σημαντικότερα πλεονεκτήματά της είναι ότι είναι διαθέσιμη σε πολλά λειτουργικά συστήματα, συνδέεται με πολλά γνωστά συστήματα βάσεων δεδομένων, έχει πολλές ενσωματωμένες βιβλιοθήκες και είναι εύκολη στην εκμάθησή της. [134]

3.4.3 Βάση δεδομένων MySQL

Η **MySQL** είναι ένα σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων που δίνει τη δυνατότητα της αποθήκευσης, αναζήτησης, ταξινόμησης, ομαδοποίησης, ανάκλησης δεδομένων με βάση τη γλώσσα ερωτημάτων SQL. Το γεγονός ότι η MySQL είναι σχεσιακή συνεπάγεται ότι η οργάνωση των δεδομένων γίνεται σε διαφορετικούς πίνακες οι οποίοι σχετίζονται μεταξύ τους με κάποιο σαφώς ορισμένο τρόπο. Επιπλέον, η MySQL μπορεί

να ελέγχει την πρόσβαση στα δεδομένα, εξασφαλίζοντας έτσι τη δυνατότητα η πρόσβαση να γίνεται από διαφορετικούς χρήστες. Κάθε χρήστης έχει συγκεκριμένα δικαιώματα πάνω στις βάσεις δεδομένων που του τα δίνει η MySQL. [2]

3.4.4 Active Record Pattern

Σε αυτό το σημείο θα πρέπει να αναφερθεί ότι πέρα από τις παραπάνω τεχνολογίες εφαρμόζεται και το μοτίβο αρχιτεκτονικής **Active Record**. Το μοτίβο αυτό είναι σχεδιασμένο με αντικειμενοστραφή τρόπο και χρησιμοποιεί ειδικές PHP κλάσεις οι οποίες περιλαμβάνουν:

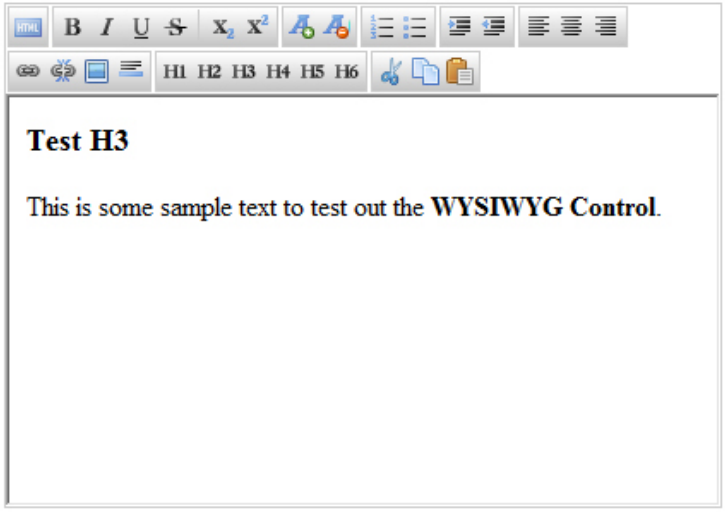
- ✓ ιδιότητες (properties), που στοχεύουν απευθείας σε κάθε ένα από τα πεδία της βάσης δεδομένων όπως επίσης και
- ✓ μεθόδους (methods) για την εισαγωγή, ανάκτηση και διαγραφή εγγραφών από την βάση δεδομένων. [133] [137]

3.4.5 Bootstrap Framework

Το **Bootstrap** είναι ένα πολύ καλό και εύχρηστο CSS framework το οποίο χρησιμοποιείται για την δημιουργία ιστοσελίδων. Ο βασικός λόγος που χρησιμοποιείται είναι για να εξασφαλιστεί ότι το περιεχόμενο της ιστοσελίδας θα εμφανίζεται με τον καλύτερο δυνατό τρόπο σε οποιαδήποτε συσκευή (και ειδικά στα κινητά τηλέφωνα) αποφεύγοντας έτσι προβλήματα ασυμβατότητας. [135]

3.4.6 **Επεξεργαστής** WYSIWYG

Για την εύκολη επεξεργασία του HTML κώδικα χρησιμοποιείται ένας πολύ απλός What You See Is What You Get (WYSIWYG) επεξεργαστής. Πρόκειται για τον επεξεργαστή **jHtmlArea** ο οποίος παρέχετε δωρεάν, είναι επεκτάσιμος και υποστηρίζεται από όλους τους περιηγητές. [138]



Εικόνα 3.24 - What You See Is What You Get Editor

3.4.7 **Ομαλοποιημένα** URL

Ένα από τα πιο σημαντικά πράγματα που οφείλει να παρέχει μια ιστοσελίδα είναι τα **ομαλοποιημένα URL (Search Engine Friendly ή SEF URLs)**. Τα SEF Url, γνωστά και ως **human-readable** ή **clean URLs**, ακολουθούν ένα προκαθορισμένο μοτίβο που καθορίζεται από το αρχείο .htaccess. [139]

Το .htaccess αρχείο που χρησιμοποιεί το **Joomla CMS** είναι κατάλληλα διαμορφωμένο και δεν χρειάζεται επιπλέον επεξεργασία. Επιπροσθέτως, μπορεί να χρησιμοποιηθεί δωρεάν και από οποιοδήποτε αφού το Joomla CMS εμπίπτει στην κατηγορία του ανοιχτού λογισμικού. [140]

3.4.8 **Χρήση** SSL (Secure Socket Layer)

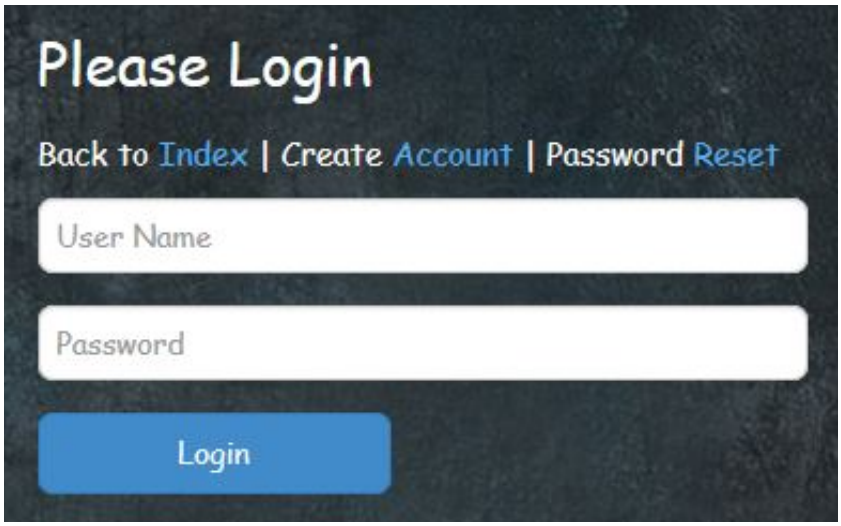
Για την ασφαλή μεταφορά των δεδομένων μεταξύ των χρηστών και της εφαρμογής χρησιμοποιείται το **GeoTrust RapidSSL** το οποίο είναι χαμηλό σε κόστος και έχει όλα τα πλεονεκτήματα που έχουν τα πρωτόκολλα που εντάσσονται στην κατηγορία **Domain Validation**. [141]

3.5 Ασφαλείς μηχανισμοί

Στο κείμενο που ακολουθεί περιγράφονται όλοι οι μηχανισμοί ασφαλείας που χρησιμοποιούνται για την προστασία τόσο του χρήστη όσο και της εφαρμογής.

3.5.1 Χρήση διαπιστευτηρίων και κρυπτογραφία

Προκειμένου ένας χρήστης να πιστοποιήσει την ταυτότητά του μπορεί να χρησιμοποιήσει τον αντίστοιχο μηχανισμό σύνδεσης (login).



Εικόνα 3.25 Login Form

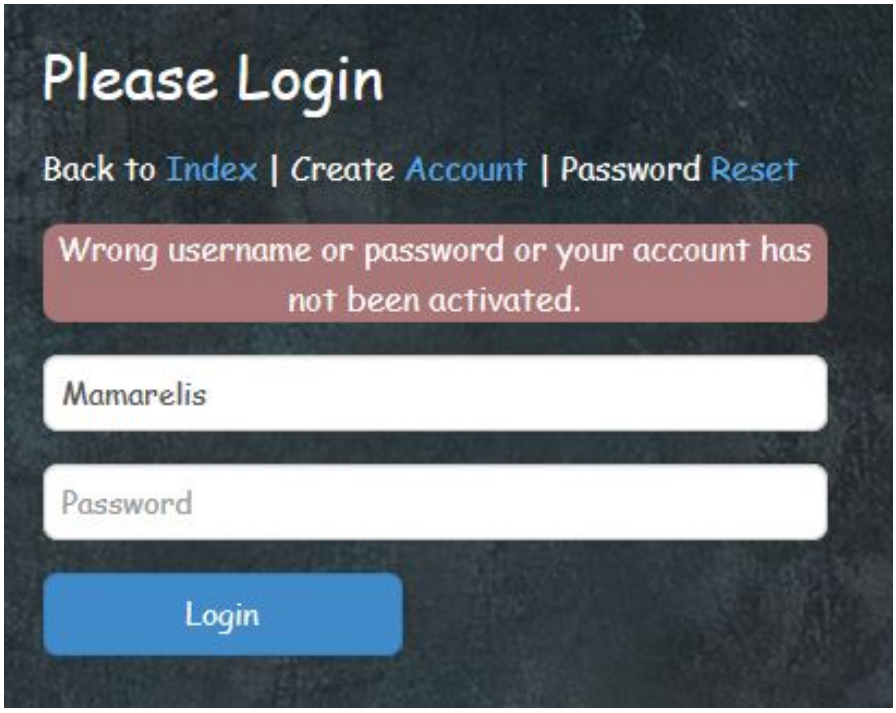
Με βάση τα διαπιστευτήρια (username & password) που εισάγει ο χρήστης γίνονται και οι αντίστοιχοι έλεγχοι. Σε περίπτωση που το *όνομα χρήστη υπάρχει* στη Βάση Δεδομένων η εφαρμογή ελέγχει αν ο κωδικός πρόσβασης είναι σωστός. Ο έλεγχος του κωδικού πρόσβασης γίνεται με τον εξής τρόπο:

1. Η εφαρμογή κρυπτογραφεί τον κωδικό πρόσβασης που έχει εισάγει ο χρήστης με χρήση του αλγόριθμου SHA256 και επιστρέφει το αποτέλεσμα σε μια μεταβλητή με όνομα $password.

2. Με βάση το όνομα χρήστη επιλέγεται μια μοναδική αλφαριθμητική τιμή από τη βάση δεδομένων η οποία αποθηκεύεται σε μια μεταβλητή με όνομα $salt

3. Στη συνέχεια, δημιουργείται ένα νέο αλφαριθμητικό το οποίο αποτελείται από τις δύο προηγούμενες μεταβλητές ($password και $salt), το οποίο κρυπτογραφείται με χρήση του αλγόριθμου SHA256 και συγκρίνεται με την αποθηκευμένη τιμή που βρίσκεται στη Βάση δεδομένων και αντιστοιχεί στο συγκεκριμένο όνομα χρήστη (username).

Αν ο χρήστης εισάγει σωστά τα διαπιστευτήρια του μπορεί να συνδεθεί και να αποκτήσει πρόσβαση σε λειτουργίες και περιεχόμενο που ένας απλός επισκέπτης δεν είναι σε θέση να έχει. Σε κάθε άλλη περίπτωση, θα εμφανιστεί κάποιο μήνυμα λάθους. [142]

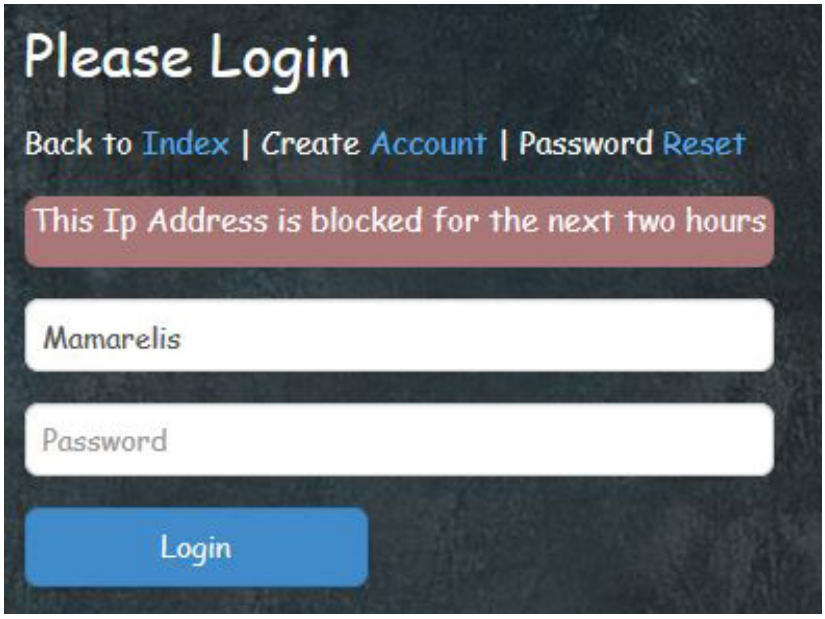


Εικόνα 3.26 - Wrong username or password

3.5.2 **Προστασία από επιθέσεις ωμής βίας (**Brute Force)

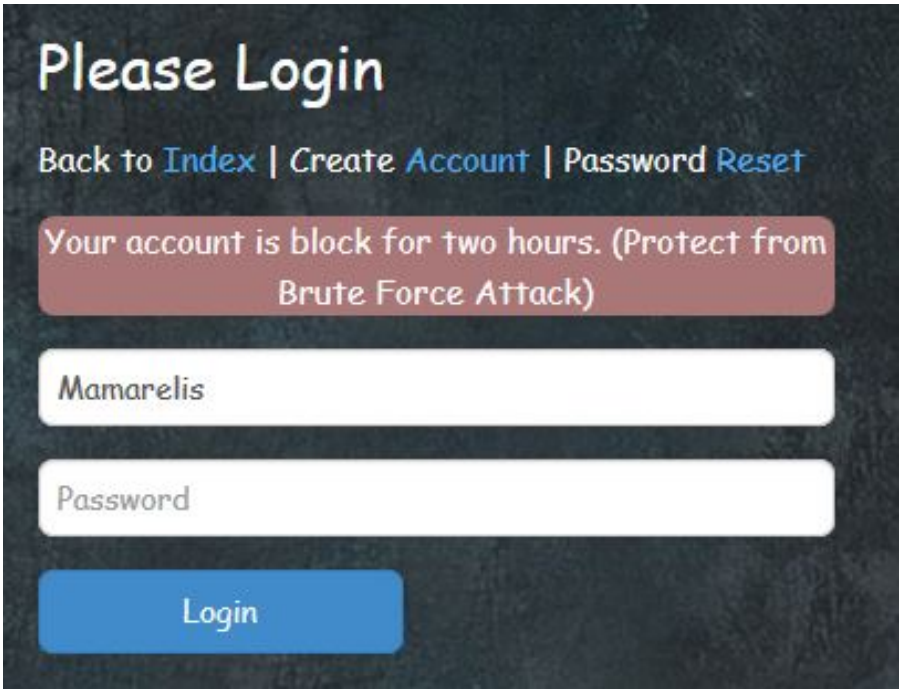
Όταν ο μηχανισμός σύνδεσης ελέγξει τα διαπιστευτήρια που εισάγει ο χρήστης και εντοπίσει ότι είναι λανθασμένα τότε ενεργοποιείται αυτόματα ο μηχανισμός προστασίας κατά των **επιθέσεων ωμής βίας (Brute Force Attacks)**. Ο μηχανισμός αυτός εξετάζει αν το όνομα χρήστη υπάρχει στη Βάση Δεδομένων και ενεργεί ανάλογα.

Σε περίπτωση που το *όνομα χρήστη δεν υπάρχει* ο μηχανισμός αποθηκεύει τη διεύθυνση IP. Αν οι αποτυχημένες προσπάθειες είναι περισσότερες από πέντε, ο μηχανισμός σύνδεσης δεν επιτρέπει σε οποιονδήποτε χρήστη να συνδεθεί, από τη συγκεκριμένη διεύθυνση IP για τις επόμενες δύο ώρες.



Εικόνα 3.27 - Ip Address is Blocked

Αν το *όνομα χρήστη είναι αποθηκευμένο* στη βάση δεδομένων αλλά ο κωδικός πρόσβασης είναι λανθασμένος τότε ο μηχανισμός μετράει τις αποτυχημένες προσπάθειες που κάνει ο χρήστης για να συνδεθεί. Αν οι αποτυχημένες προσπάθειες είναι περισσότερες από πέντε τότε ο μηχανισμός δεν επιτρέπει στο συγκεκριμένο χρήστη να συνδεθεί από οποιαδήποτε IP διεύθυνση για τις επόμενες δύο ώρες. [143] [144] [145]



Εικόνα 3.28 - Protect from Brute Force Attack

3.5.3 **Διαχείριση συνόδων** (Session Management)

Για λόγους ασφαλείας το αναγνωριστικό συνόδου (session ID) αναπαράγεται μετά από κάθε αίτημα (request). Όλα τα δεδομένα των συνόδων κρυπτογραφούνται και αποθηκεύονται μέσα στη βάση δεδομένων και όχι μέσα στον εξυπηρετητή. Με αυτό τον τρόπο είναι πολύ πιο δύσκολο για τον εισβολέα να πραγματοποιήσει οποιαδήποτε τεχνική επίθεσης σχετίζεται με την **πρόβλεψη διαπιστευτηρίων / συνόδου (Credential / Session Prediction)**. [146] [147]

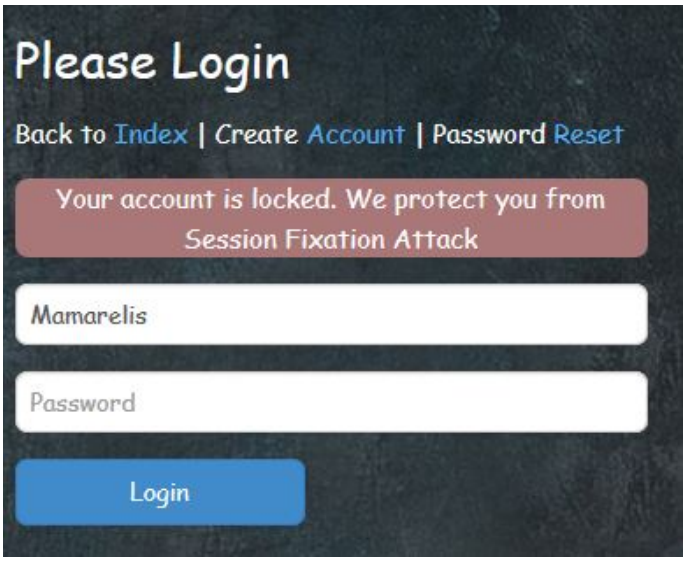
3.5.4 **Αποσύνδεση χρήστη (**Log Out)

Ένας συνδεδεμένος χρήστης (logged In) μπορεί να αποσυνδεθεί (Log Out) από την εφαρμογή όταν αυτός θελήσει χρησιμοποιώντας την αντίστοιχη λειτουργία.

Εικόνα 3.29 - User Control Module

Εναλλακτικά, ο μηχανισμός αποσύνδεσης μπορεί να ενεργοποιηθεί μετά το προκαθορισμένο όριο αδράνειας (το οποίο είναι διαφορετικό σε κάθε εφαρμογή). Στη συγκεκριμένη εφαρμογή γίνεται χρήση μιας χρονοσφραγίδας (timestamp) η οποία ενεργοποιεί το μηχανισμό αποσύνδεσης όταν το χρονικό διάστημα μεταξύ δύο αιτημάτων (requests) είναι μεγαλύτερο των τριάντα λεπτών. Σε αυτή την περίπτωση ο χρήστης καλείται να εισάγει ξανά τα διαπιστευτήρια του προκειμένου να πιστοποιήσει και πάλι την ταυτότητά του.

Η μόνη περίπτωση κατά την οποία ενεργοποιείται ο μηχανισμός αποσύνδεσης και ταυτόχρονα ο λογαριασμός ενός χρήστη απενεργοποιείται είναι να έχουν συνδεθεί ταυτόχρονα στην εφαρμογή, με το ίδιο όνομα χρήστη (username), δύο ή περισσότεροι χρήστες. Με αυτό τον τρόπο προστατεύεται ο λογαριασμός ενός χρήστη από μια επίθεση κακόβουλου ορισμού αναγνωριστικών συνόδου (Session Fixation). Για τον σκοπό αυτό η εφαρμογή αποθηκεύει το αποτύπωμα του περιηγητή (browser fingerprint) του χρήστη στην βάση δεδομένων και ελέγχει αν είναι το ίδιο σε κάθε αίτημα (request). [148] [149]

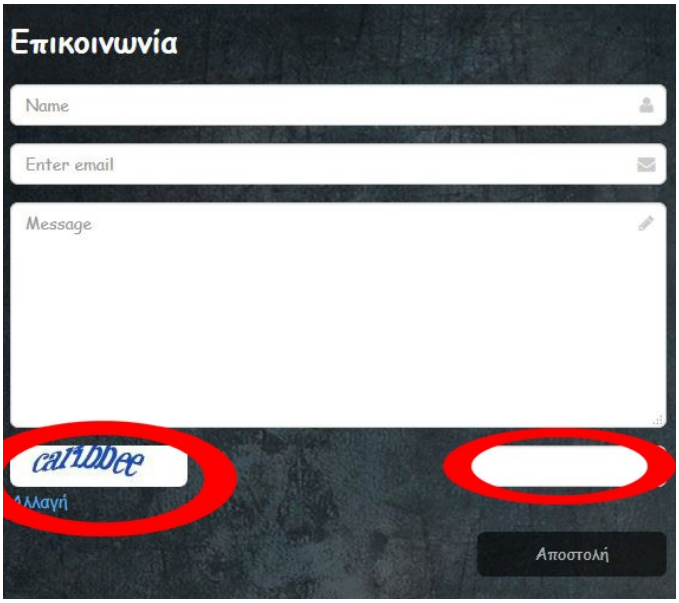


Εικόνα 3.30 - Protect from Session Fixation Attack

3.5.5 **Χρήση** Captcha

Το **Captcha (Completely Automated Public Turing test to tell Computers and Humans Apart)** είναι ένας μηχανισμός που χρησιμοποιείται για να εξασφαλίσει ότι τα πεδία μιας φόρμας έχουν συμπληρωθεί από κάποιον χρήστη. Χρησιμοποιείται από το 2000 συμβάλλοντας αποτελεσματικά στην αντιμετώπιση του spamming καθώς επίσης και στην αντιμετώπιση επιθέσεων ωμής βίας (Brute Force).

Το Captcha αποτελεί μέρος μιας φόρμας και απαρτίζεται από μια εικόνα που παρουσιάζει συνδυασμούς γραμμάτων και αριθμών σε διάφορα μεγέθη, χρώματα κλπ. και από ένα υποχρεωτικό πεδίο στο οποίο καλείται ο χρήστης να εισάγει αυτό το συνδυασμό. [150]



Εικόνα 3.31 - Captcha

3.5.6 **Εφαρμογή** SSL (Secure Socket Layer)

Η σωστή χρήση του πρωτόκολλου ασφαλείας **SSL (Secure Socket Layer)** μπορεί να προστατέψει μια εφαρμογή από πάρα πολλές επιθέσεις. Αυτό συνεπάγεται ότι το πρωτόκολλο SSL θα πρέπει να χρησιμοποιείται σε όλες τις σελίδες της εφαρμογής. Η χρήση του SSL σε ορισμένες σελίδες είναι μια λανθασμένη τακτική που δημιουργεί κενά ασφαλείας. Ένας τρόπος για να χρησιμοποιηθεί σωστά, είναι η χρήση των κατάλληλων εντολών μέσα από το αρχείο .htaccess.

Η σωστή χρήση του πρωτόκολλου SSL προστατεύει την εφαρμογή από την ευπάθεια της **ανεπαρκής προστασίας επιπέδου μεταφοράς (Insufficient Transport Layer Protection)** και παράλληλα συμβάλει στην αντιμετώπιση επιθέσεων όπως: [33]

- ✓ Στην πρόβλεψη διαπιστευτηρίων / συνόδου (Credential /Session Prediction)
- ✓ Στα λαθραία αιτήματα HTTP (HTTP Request Smuggling)
- ✓ Στη διάσπαση αιτήσεων HTTP (HTTP Request Splitting)
- ✓ Στη διάσπαση απαντήσεων HTTP (HTTP Response Splitting)
- ✓ Στον κακόβουλο ορισμό αναγνωριστικών συνόδου (Session Fixation)

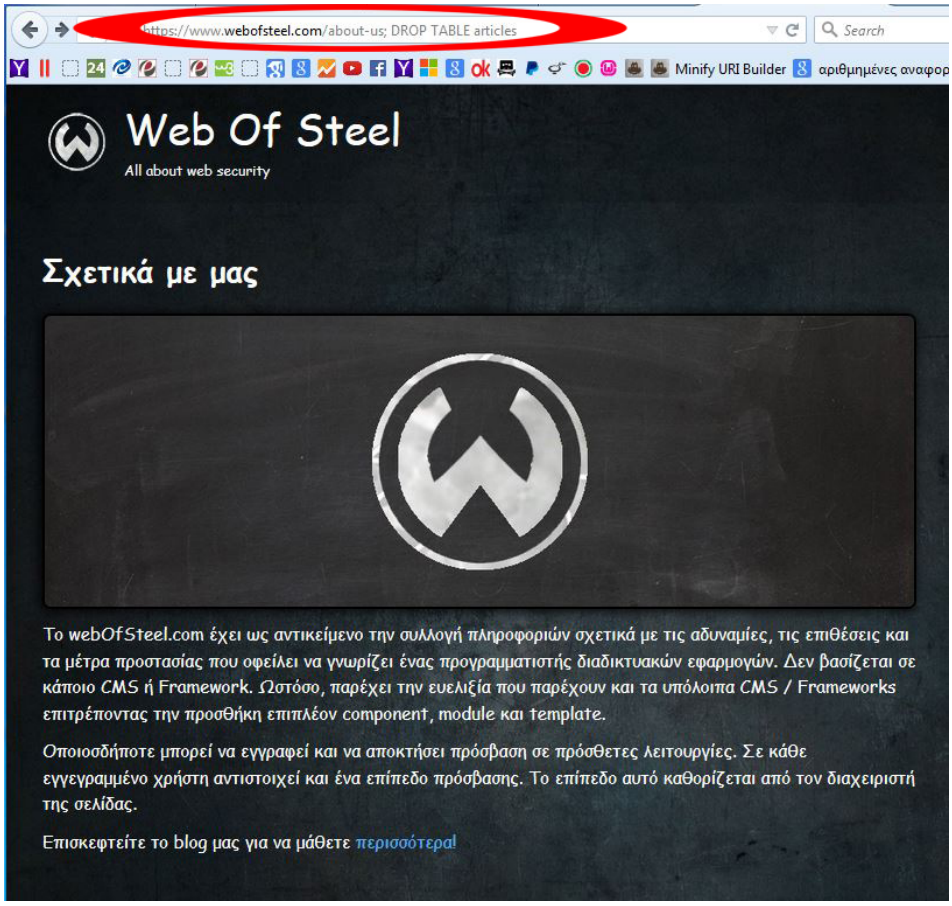
3.5.7 **Προστασία από επιθέσεις έγχυσης κώδικα** (Code Injections)

Με τον όρο επίθεση έγχυσης κώδικα (code Injection) συμπεριλαμβάνονται όλες οι επιθέσεις που βασίζονται στην ευπάθεια **του ακατάλληλου χειρισμού εισόδου και εξόδου (Improper Input & output Handling)** των δεδομένων που εισάγουν οι χρήστες στην εφαρμογή. Όλες οι φόρμες και τα πιθανά σημεία εισόδου και εξόδου δεδομένων έχουν σχεδιαστεί με το σκεπτικό ότι το περιεχόμενο που εισάγει οποιοσδήποτε χρήστης είναι ύποπτο. Για το λόγο αυτό ακολουθούνται τεχνικές **επικύρωσης (validation)** και **καθαρισμού (sanitizing)** των δεδομένων που πρόκειται να εισαχθούν στην εφαρμογή καθώς επίσης και διαδικασίες **διαφυγής (escaping)** και **κωδικοποίησης (encode)** των δεδομένων που πρόκειται να εξαχθούν από αυτή.

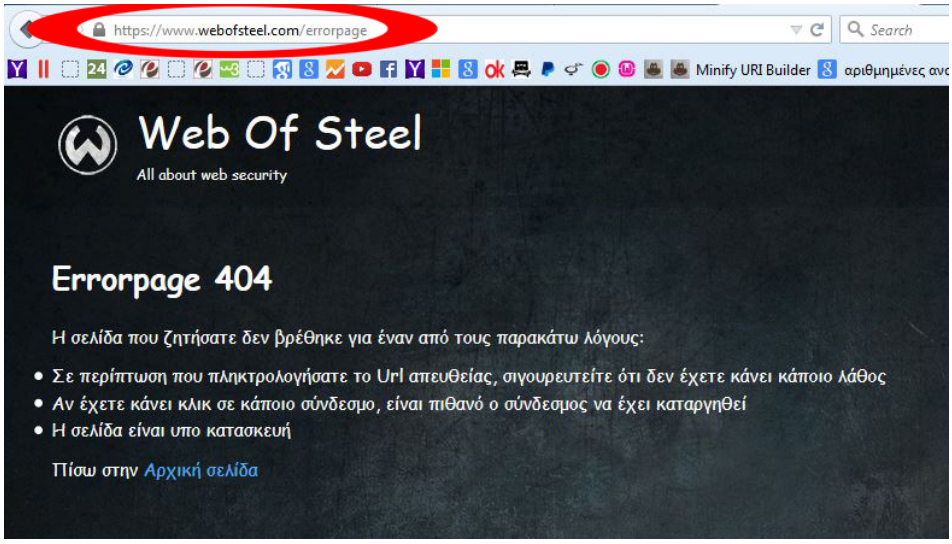
3.5.7.1 **Προστασία από επιθέσεις έγχυσης σε εντολές** SQL (SQL Injections)

Οι επιθέσεις έγχυσης σε εντολές SQL είναι οι πιο γνωστές επιθέσεις έγχυσης κώδικα (code Injection). Ως εκ τούτου, η εφαρμογή είναι διαμορφωμένη με τέτοιο τρόπο που να μπορεί να αποτρέψει αυτή την επίθεση.

Για την ασφάλεια του χρήστη και της εφαρμογής ελέγχεται το περιεχόμενο που εισάγουν οι χρήστες τόσο στις διάφορες φόρμες όσο και στη μπάρα διεύθυνσης του περιηγητή.



Εικόνα 3.32 - Prevent SQL Injection (Url Handling)



Εικόνα 3.33 Errorpage 404 Component

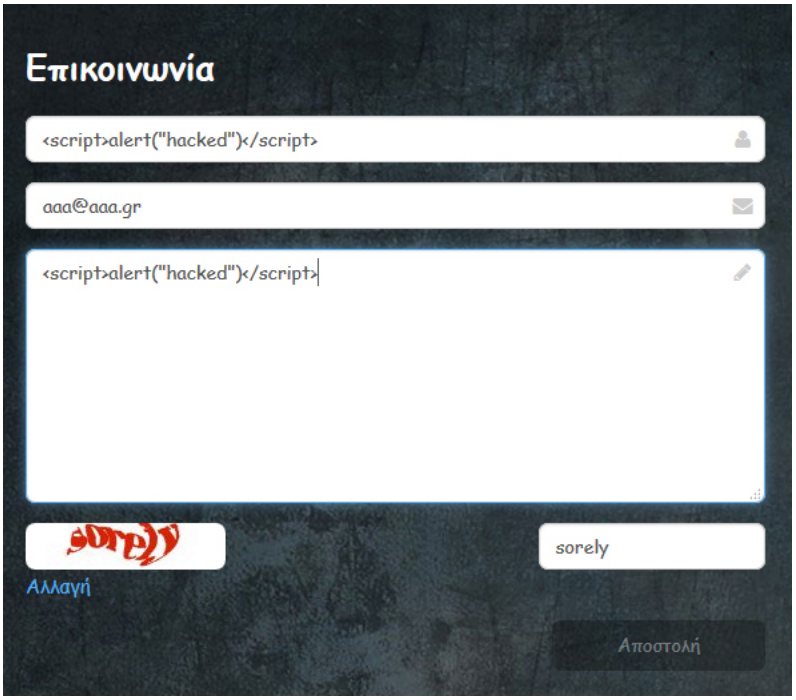
Εικόνα 3.34 Prevent SQL Injection (Search Bar)

3.5.7.2 **Προστασία από επιθέσεις** Cross-Site Scripting (XSS)

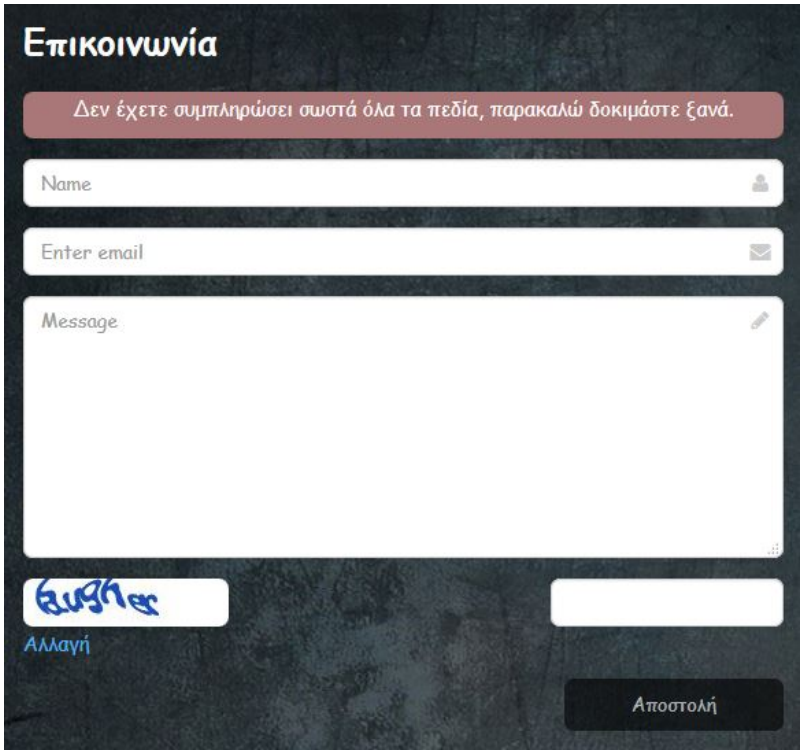
Η εφαρμογή είναι σε θέση να ελέγχει όλα τα σημεία εισόδου από τα οποία μπορεί ένας χρήστης να εισάγει κάποιο κακόβουλο script (σε μορφή JavaScript όπως π.χ. <script>alert('Hacked');</script>) ή ακόμα και HTML Tags.

Εικόνα 3.35 Remove Html Tags (Search Bar Module)

Εικόνα 3.36 - Prevent Cross-Site Scripting (XSS) (Search Bar Module)



Εικόνα 3.37 - Prevent Cross-site Scripting (XSS) (1)



Εικόνα 3.38 - Prevent Cross-site Scripting (XSS) (2)

4. Συμπεράσματα

Οι ειδικοί σε θέματα ασφάλειας παρομοιάζουν το διαδίκτυο με μια γειτονιά υψηλής εγκληματικότητας, όπου θα πρέπει να έχει κανείς παράθυρα και πόρτες κλειδωμένα, μένοντας πάντα με τα μάτια ανοιχτά για να κοιτάει παντού. Είναι το μέρος που αναμένει κανείς να πέσει συχνά θύμα κλοπής. Το γεγονός αυτό οφείλεται τόσο στο τρόπο με τον οποίο μπορεί να έχει δομήσει κάποιος μια διαδικτυακή εφαρμογή, όσο και στον τρόπο με τον οποίο είναι σχεδιασμένο το ίδιο το διαδίκτυο.

Το διαδίκτυο που όλοι γνωρίζουμε σήμερα δεν έχει καμία σχέση με το διαδίκτυο των αρχών του 1990, το οποίο ήταν ένα πολύ ασφαλές μέσο επικοινωνίας και ανταλλαγής πληροφοριών. Δυστυχώς, το διαδίκτυο πλέον αποτελεί μια από τις λιγότερο ασφαλείς εφευρέσεις που κατασκεύασε ποτέ ο άνθρωπος. Μέσα σε λίγα μόλις χρόνια το διαδίκτυο μετατράπηκε σε ένα τεράστιο εμπορικό κέντρο, χωρίς τις σωστές προδιαγραφές, γεγονός που οφείλεται στην απληστία των ανθρώπων.

Κανένας δεν μπορεί να πει ότι στο διαδίκτυο κάποιος είναι απόλυτα ασφαλής άλλα ούτε και το αντίθετο. Η ασφάλεια εξαρτάται από όλους όσους εμπλέκονται με τη χρήση του διαδικτύου. Από την μια πλευρά, οι προγραμματιστές διαδικτυακών εφαρμογών οφείλουν να χρησιμοποιούν σωστά τα μέτρα προστασίας και να κάνουν τους απαραίτητους ελέγχους προκειμένου να εντοπίσουν τα κενά ασφαλείας που ενδεχομένως υπάρχουν στην εφαρμογή και στον εξυπηρετητή τους. Από την άλλη, οι χρήστες οφείλουν να είναι εξίσου ενημερωμένοι για τους κινδύνους που υπάρχουν για να μην βρεθούν εκτεθειμένοι.

Όσον αφορά την εφαρμογή, έχει ελεγχθεί σε αρκετές επιθέσεις και περιλαμβάνει τους απαραίτητους ελέγχους στα σημεία όπου γίνεται εισαγωγή και εξαγωγή των δεδομένων που έχουν εισάγει οι χρήστες. Επιπροσθέτως, έχει εγκατασταθεί πιστοποιητικό SSL για την ασφαλή μεταφορά των δεδομένων που ανταλλάσσονται μεταξύ των χρηστών και της εφαρμογής. Ωστόσο, κάθε φόρα που πρόκειται να προστεθεί οτιδήποτε καινούριο (component, module ή template) θα πρέπει να πραγματοποιούνται εκ νέου όσο το δυνατόν περισσότεροι έλεγχοι ασφαλείας είναι εφικτό.

Η συχνή και σωστή ενημέρωση είναι απαραίτητη για τον εντοπισμό αδυναμιών και την αποτελεσματική πρόληψη των ενδεχόμενων επιθέσεων. Είναι αδύνατον για οποιοδήποτε προγραμματιστή διαδικτυακών εφαρμογών να μάθει να αμύνεται αν πρωτίστως δεν έχει μάθει πως να επιτίθεται.

5. Μελλοντικές επεκτάσεις

Η εφαρμογή αυτή είναι σχεδιασμένη με τέτοιο τρόπο που ευνοεί τη βελτίωση τόσο σε επίπεδο λειτουργικότητας όσο και σε επίπεδο ασφάλειας. Πιο συγκεκριμένα η εφαρμογή ευνοεί:

- Το σχεδιασμό και τη προσθήκη απεριόριστων πρόσθετων (component, module και template), όπως συμβαίνει και με τα υπόλοιπα συστήματα διαχείρισης περιεχομένου (Content Management Systems / CMS).

- Τη προσθήκη ενός γραφικού περιβάλλοντος (backend), ακριβώς όπως συμβαίνει και με υπόλοιπα συστήματα διαχείρισης περιεχομένου (CMS), το οποίο θα δίνει τη δυνατότητα στο χρήστη να διαχειριστεί και να ελέγχει όλα τα πρόσθετα στοιχεία (component, module και template) με τον τρόπο που επιθυμεί αυτός.

- Τη δυνατότητα μετατροπής της εφαρμογής σε μια πολύγλωσση ιστοσελίδα.

- Τη χρήση της γλώσσας XML (eXtensible Markup Language) σε παλιές και νέες λειτουργίες.

- Τη προσθήκη ενός μηχανισμού που θα επιτρέπει στους χρήστες να χρησιμοποιούν μάρκες ασφαλείας (Security Tokens) κατά τη διαδικασία σύνδεσης (Log In).

- Τη προσθήκη ενός μηχανισμού που θα επιτρέπει στους χρήστες να συνδεθούν μέσα από ένα εικονικό δίκτυο περιήγησης (Virtual Private Network / VPN) με συγκεκριμένη διεύθυνση IP και πρωτόκολλο ασφαλείας.

6 Βιβλιογραφία

1. el.wikipedia.org (2014), Διαδίκτυο.
   Πηγή από το διαδίκτυο: http://el.wikipedia.org/wiki/Διαδίκτυο
   Ανακτήθηκε στις 2 Ιανουαρίου 2015.

2. Κούρτης Νικόλαος (2009), Διπλωματική εργασία: Κατασκευή Ιστοσελίδας Ηλεκτρονικού Καταστήματος με χρήση Php - MySQL - Apache,
   Τμήμα ηλεκτρολόγων μηχανικών & τεχνολογίας υπολογιστών,
   Πανεπιστήμιο Πατρών

3. el.wikipedia.org (2014), Διαδικτυακή εφαρμογή
   Πηγή από το διαδίκτυο: http://el.wikipedia.org/wiki/Διαδικτυακή_εφαρμογή
   Ανακτήθηκε στις 2 Ιανουαρίου 2015.

4. Πάσχου Μερσίνη (2009), Διπλωματική εργασία: Αποδοτικές Τεχνικές Διαχείρισης Εφαρμογών και Υπηρεσιών Διαδικτύου (Web Services) σε Πληροφοριακά Συστήματα Ηλεκτρονικής Διακυβέρνησης (e- goverment),
   Τμήμα μηχανικών ηλεκτρονικών Υπολογιστών & Πληροφορικής,
   Πανεπιστήμιο Πατρών

5. Μπαλαφούτης Χρήστος (2012), Διπλωματική Εργασία: Μέθοδοι προστασίας ιστοσελίδων στο διαδίκτυο,
   Τμήμα ηλεκτρολόγων μηχανικών και τεχνολογίας Υπολογιστών,
   Πανεπιστήμιο Πατρών

6. en.wikipedia.org (2014), Multitier architecture.
   Πηγή από το διαδίκτυο: http://en.wikipedia.org/wiki/Multitier_architecture
   Ανακτήθηκε στις 2 Ιανουαρίου 2015.

7. Μακρυπόδης Βασίλειος (2013), Μεταπτυχιακή εργασία: Έλεγχος ασφάλειας ιστοχώρων: μεθοδολογίες και έλεγχος ευπαθειών,
   Τμήμα Πληροφορικής και Τηλεπικοινωνιών,
   Πανεπιστήμιο Πελοποννήσου

8. Ανδρέας Κ. Μάττας (2007), Διδακτορική Διατριβή: Ασφάλεια Πληροφοριακών συστημάτων σε συνεργατικά περιβάλλοντα εφαρμογών με βάση το διαδίκτυο,
   Τομέας Υπολογιστικών μεθόδων και προγραμματισμού Η/Υ, Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης

9. Αλεξανδροπούλου Μαρία (2011), Μεταπτυχιακή Εργασία: Μέθοδοι αξιολόγησης της ασφάλειας λογισμικού εφαρμογών Τμήμα επιστήμης και Τεχνολογίας Υπολογιστών, Πανεπιστήμιο Πελοποννήσου

10. Λουκία Ι. Τζιοβάνη, Διπλωματική Εργασία: Ασφάλεια στο Ηλεκτρονική Εμπόριο Σχολή ηλεκτρολόγων μηχανικών και μηχανικών υπολογιστών, Τομέας συστημάτων μετάδοσης πληροφορίας και τεχνολογίας Υλικών, Εθνικό Μετσόβιο Πολυτεχνείο

11. projects.webappsec.org (2010), Application Misconfiguration Πηγή από το διαδίκτυο: http://projects.webappsec.org/w/page/13246914/Application Misconfiguration Ανακτήθηκε στις 1 Ιουνίου 2014.

12. projects.webappsec.org (2010), Directory Indexing Πηγή από το διαδίκτυο: http://projects.webappsec.org/w/page/13246922/Directory Indexing Ανακτήθηκε στις 1 Ιουνίου 2014.

13. projects.webappsec.org (2010), Πηγή από το διαδίκτυο: http://projects.webappsec.org/w/page/13246932/Improper Filesystem Permissions Ανακτήθηκε στις 1 Ιουνίου 2014.

14. terrychay.com (2007), Filter Input-Escape Output: Security Principle and Practice Πηγή από το διαδίκτυο: http://terrychay.com/article/php-advent-security-filter-input-escape-output.shtml Ανακτήθηκε στις 1 Ιουνίου 2014

15. projects.webappsec.org (2010), Improper Input Handling Πηγή από το διαδίκτυο: http://projects.webappsec.org/w/page/13246933/Improper Input Handling Ανακτήθηκε στις 1 Ιουνίου 2014

16. projects.webappsec.org (2010), Improper Output Handling Πηγή από το διαδίκτυο: http://projects.webappsec.org/w/page/13246934/Improper Output Handling Ανακτήθηκε στις 1 Ιουνίου 2014

17. projects.webappsec.org (2010), Information Leakage Πηγή από το διαδίκτυο: http://projects.webappsec.org/w/page/13246936/Information Leakage Ανακτήθηκε στις 1 Ιουνίου 2014

18. projects.webappsec.org (2010), Insecure Indexing
Πηγή από το διαδίκτυο:
http://projects.webappsec.org/w/page/13246937/Insecure Indexing
Ανακτήθηκε στις 1 Ιουνίου 2014

19. projects.webappsec.org (2010), Insufficient Anti-automation
Πηγή από το διαδίκτυο:
http://projects.webappsec.org/w/page/13246938/Insufficient Anti-automation
Ανακτήθηκε στις 1 Ιουνίου 2014

20. projects.webappsec.org (2010), Insufficient Authentication
Πηγή από το διαδίκτυο:
http://projects.webappsec.org/w/page/13246939/Insufficient Authentication
Ανακτήθηκε στις 1 Ιουνίου 2014

21. projects.webappsec.org (2010), Insufficient Authorization
Πηγή από το διαδίκτυο:
http://projects.webappsec.org/w/page/13246940/Insufficient Authorization
Ανακτήθηκε στις 1 Ιουνίου 2014

22. projects.webappsec.org (2010), Insufficient Password Recovery
Πηγή από το διαδίκτυο:
http://projects.webappsec.org/w/page/13246942/Insufficient Password Recovery
Ανακτήθηκε στις 1 Ιουνίου 2014

23. projects.webappsec.org (2010), Insufficient Process Validation
Πηγή από το διαδίκτυο:
http://projects.webappsec.org/w/page/13246943/Insufficient Process Validation
Ανακτήθηκε στις 1 Ιουνίου 2014

24. projects.webappsec.org (2010), Insufficient Session Expiration
Πηγή από το διαδίκτυο:
http://projects.webappsec.org/w/page/13246944/Insufficient Session Expiration
Ανακτήθηκε στις 1 Ιουνίου 2014

25. projects.webappsec.org (2014), Insufficient Transport Layer Protection
Πηγή από το διαδίκτυο:
http://projects.webappsec.org/w/page/13246945/Insufficient Transport Layer Protection
Ανακτήθηκε στις 1 Ιουνίου 2014

26. Veracode.com, Insufficient transport layer protection
Πηγή από το διαδίκτυο: http://www.veracode.com/security/insufficient-transport-layer-protection Ανακτήθηκε στις 1 Ιουνίου 2014

27. projects.webappsec.org (2014), Server Misconfiguration
Πηγή από το διαδίκτυο: http://projects.webappsec.org/w/page/13246959/Server Misconfiguration
Ανακτήθηκε στις 1 Ιουνίου 2014

28. infosecpro.com, Abuse of Functionality
Πηγή από το διαδίκτυο: http://www.infosecpro.com/applicationsecurity/a61.htm
Ανακτήθηκε στις 1 Ιουνίου 2014

29. whitehatsec.com, Website Security Glossary Business Logic Vulnerabilities
Πηγή από το διαδίκτυο: https://www.whitehatsec.com/resource/glossary.html
Ανακτήθηκε στις 18 Ιουνίου 2014

30. el.wikipedia.org (2013), Brute-force attack
Πηγή από το διαδίκτυο: http://el.wikipedia.org/wiki/Brute-force_attack
Ανακτήθηκε στις 18 Ιουνίου 2014

31. cs.virginia.edu, Blocking Brute Force Attacks
Πηγή από το διαδίκτυο:
http://www.cs.virginia.edu/~csadmin/gen_support/brute_force.php
Ανακτήθηκε στις 18 Ιουνίου 2014

32. techopedia.com, Brute force attack
Πηγή από το διαδίκτυο: http://www.techopedia.com/definition/18091/brute-force-attack
Ανακτήθηκε στις 18 Ιουνίου 2014

33. cyberciti.biz (2006), Apache: Redirect HTTP to HTTPS Apache secure connection - force HTTPS Connections
Πηγή από το διαδίκτυο: http://www.cyberciti.biz/tips/howto-apache-force-https-secure-connections.html
Ανακτήθηκε στις 18 Ιουνίου 2014

34. en.wikipedia.org (2014), Buffer overflow
Πηγή από το διαδίκτυο: http://en.wikipedia.org/wiki/Buffer_overflow
Ανακτήθηκε στις 18 Ιουνίου 2014

35. resources.infosecinstitute.com, Content spoofing
Πηγή από το διαδίκτυο: http://resources.infosecinstitute.com/content-spoofing/
Ανακτήθηκε στις 19 Ιουνίου 2014

36. en.wikipedia.org (2014), Session hijacking
Πηγή από το διαδίκτυο: http://en.wikipedia.org/wiki/Session_hijacking
Ανακτήθηκε στις 19 Ιουνίου 2014

37. owasp.org (2014), Session hijacking attack

Πηγή από το διαδίκτυο:
https://www.owasp.org/index.php/Session_hijacking_attack
Ανακτήθηκε στις 28 Αυγούστου 2014

38. opensourceforu.com (2010), Security Apache, Part 4: Cross-site Tracing (XST) & Cross-site History Manipulation (XSHM)
Πηγή από το διαδίκτυο: http://www.opensourceforu.com/2010/12/securing-apache-part-4-xst-xshm/
Ανακτήθηκε στις 14 Ιουνίου 2014

39. owasp.org (2010), Cross Site History Manipulation (XSHM)
Πηγή από το διαδίκτυο:
https://www.owasp.org/index.php/Cross_Site_History_Manipulation_(XSHM)
Ανακτήθηκε στις 19 Ιουνίου 2014

40. Alex Roichman (2010), Cross Site History Manipulation: XSHM
Checkmark Research Labs

41. infosectoday.com, Vulnerability Case Study: Cookie Tampering
Πηγή από το διαδίκτυο:
http://www.infosectoday.com/Articles/Cookie_Tampering.htm
Ανακτήθηκε στις 21 Ιουνίου 2014

42. securiteam.com (2006), Cross Site Cooking
Πηγή από το διαδίκτυο:
http://www.securiteam.com/securityreviews/5EP0L2KHFG.html
Ανακτήθηκε στις 28 Μαρτίου 2014

43. en.wikipedia.org (2014),Cross-site scripting
Πηγή από το διαδίκτυο: http://en.wikipedia.org/wiki/Cross-site_scripting
Ανακτήθηκε στις 18 Ιουνίου 2014

44. veracode.com, Cross-Site Scripting (XSS) Tutorial: Learn About XSS Vulnerabilities, Injections and How to Prevent Attacks
Πηγή από το διαδίκτυο: http://www.veracode.com/security/xss
Ανακτήθηκε στις 18 Ιουνίου 2014

45. el.wikipedia.org (2014), Cross-Site Scripting
Πηγή από το διαδίκτυο: http://el.wikipedia.org/wiki/Cross-site_scripting
Ανακτήθηκε στις 8 Ιουνίου 2014

46. sitepoint.com (2012),Cross-site scripting attacks (XSS)
Πηγή από το διαδίκτυο: http://www.sitepoint.com/php-security-cross-site-scripting-attacks-xss/
Ανακτήθηκε στις 8 Ιουνίου 2014

47. en.wikipedia.org (2014), Computer worm
Πηγή από το διαδίκτυο: http://en.wikipedia.org/wiki/Computer_worm
Ανακτήθηκε στις 8 Ιουνίου 2014

48. en.wikipedia.org (2014), Cross-site request forgery
Πηγή από το διαδίκτυο: http://en.wikipedia.org/wiki/Cross-site_request_forgery
Ανακτήθηκε στις 11 Ιουνίου 2014

49. sitepoint.com (2011), Preventing cross-site request forgeries (CSRF)
Πηγή από το διαδίκτυο: http://www.sitepoint.com/preventing-cross-site-request-forgeries/
Ανακτήθηκε στις 11 Ιουνίου 2014

50. owasp.org (2014), Cross-Site Request Forgery (CSRF) Prevention Cheat Sheet
Πηγή από το διαδίκτυο: https://www.owasp.org/index.php/Cross-Site_Request_Forgery_(CSRF)_Prevention_Cheat_Sheet
Ανακτήθηκε στις 11 Ιουνίου 2014

51. veracode.com, Cross-Site Request Forgery Guide: Learn All About CSRF Attacks and CSRF Protection
Πηγή από το διαδίκτυο: http://www.veracode.com/security/csrf
Ανακτήθηκε στις 12 Ιουνίου 2014

52. owasp.org (2014), Cross Site Tracing
Πηγή από το διαδίκτυο: https://www.owasp.org/index.php/Cross_Site_Tracing
Ανακτήθηκε στις 14 Δεκεμβρίου 2014

53. computerweekly.com (2009), How to prevent a cross-site tracing vulnerability exploit
Πηγή από το διαδίκτυο: http://www.computerweekly.com/tip/How-to-prevent-a-cross-site-tracing-vulnerability-exploit
Ανακτήθηκε στις 11 Ιουνίου 2014

54. en.wikipedia.org (2013), Cross-zone scripting
Πηγή από το διαδίκτυο: http://en.wikipedia.org/wiki/Cross-zone_scripting
Ανακτήθηκε στις 11 Δεκεμβρίου 2014

55. minsky.gsi.dit.upm.es (2014),Cross_Zone Scripting
Πηγή από το διαδίκτυο: http://minsky.gsi.dit.upm.es/semanticwiki/index.php/Cross_Zone_Scripting
Ανακτήθηκε στις 11 Δεκεμβρίου 2014

56. owasp.org (2009), Cross-User Defacement
Πηγή από το διαδίκτυο: https://www.owasp.org/index.php/Cross-User_Defacement
Ανακτήθηκε στις 11 Δεκεμβρίου 2014

57. minsky.gsi.dit.upm.es (2014), Cross-User Defacement
Πηγή από το διαδίκτυο:
http://minsky.gsi.dit.upm.es/semanticwiki/index.php/Cross-User_Defacement
Ανακτήθηκε στις 11 Δεκεμβρίου 2014

58. en.wikipedia.org (2014), Denial-of-service attack
Πηγή από το διαδίκτυο: http://en.wikipedia.org/wiki/Denial-of-service_attack
Ανακτήθηκε στις 12 Δεκεμβρίου 2014

59. eeei.gr, Είδη επιθέσεων DoS (Denial of Service)
Πηγή από το διαδίκτυο: http://www.eeei.gr/interbiz/articles/dos.htm
Ανακτήθηκε στις 12 Δεκεμβρίου 2014

60. us-cert.gov (2013), Security Tip (ST04-015) Understanding Denial-of-service (DoS) attack
Πηγή από το διαδίκτυο: https://www.us-cert.gov/ncas/tips/ST04-015
Ανακτήθηκε στις 12 Δεκεμβρίου 2014

61. owasp.org (2014), Denial of Service
Πηγή από το διαδίκτυο: https://www.owasp.org/index.php/Denial_of_Service
Ανακτήθηκε στις 11 Δεκεμβρίου 2014

62. esecurityplanet.com (2012), How to prevent DoS Attacks
Πηγή από το διαδίκτυο: http://www.esecurityplanet.com/network-security/how-to-prevent-dos-attacks.html
Ανακτήθηκε στις 11 Δεκεμβρίου 2014

63. techopedia.com, Denial-of-Service Attack (DoS)
Πηγή από το διαδίκτυο: http://www.techopedia.com/definition/24841/denial-of-service-attack-dos
Ανακτήθηκε στις 11 Δεκεμβρίου 2014

64. Kevin Beaver (2013), Hacking For Dummies 4th Edition, John Wiley & Sons, Inc.

65. Ιωάννης Βήττα (2009), Κατασκευή Συστήματος Αναγνώρισης Κακόβουλων Χρηστών Στο Διαδίκτυο
Τμήμα Ηλεκτρολόγων Μηχανικών και Τεχνολογίας Υπολογιστών, Πανεπιστήμιο Πάτρας

66. Στυλιανός Γεώργιος (2013), Αναγνώριση επιθέσεων web σε web-servers
Τμήμα Ηλεκτρολόγων Μηχανικών και τεχνολογίας Υπολογιστών, Πανεπιστήμιο Πατρών

67. caclab.csd.auth.gr, Ασφάλεια Web Client

Πηγή από το διαδίκτυο: http://caclab.csd.auth.gr/WebSecurityClientSide.pdf
Ανακτήθηκε στις 11 Δεκεμβρίου 2014

68. gohacking.com, DNS Hijacking: What is it and How it Works
Πηγή από το διαδίκτυο: http://www.gohacking.com/dns-hijacking/
Ανακτήθηκε στις 13 Δεκεμβρίου 2014

69. thevpn.guru (2014),DNS Hijacking: Exposed & Explained
Πηγή από το διαδίκτυο: https://thevpn.guru/dns-hijacking-exposed-explained/
Ανακτήθηκε στις 13 Δεκεμβρίου 2014

70. owasp.org (2009), Cache Poisoning
Πηγή από το διαδίκτυο: https://www.owasp.org/index.php/Cache_Poisoning
Ανακτήθηκε στις 13 Δεκεμβρίου 2014

71. en.wikipedia.org (2014), DNS Spoofing
Πηγή από το διαδίκτυο: http://en.wikipedia.org/wiki/DNS_spoofing
Ανακτήθηκε στις 13 Δεκεμβρίου 2014

72. veracode.com, Cache Poisoning Attack
Πηγή από το διαδίκτυο: http://www.veracode.com/security/cache-poisoning
Ανακτήθηκε στις 14 Δεκεμβρίου 2014

73. acunetix.com, What is Google Hacking?
Πηγή από το διαδίκτυο: http://www.acunetix.com/websitesecurity/google-hacking/
Ανακτήθηκε στις 14 Δεκεμβρίου 2014

74. en.wikipedia.org (2014), HTTP header injection
Πηγή από το διαδίκτυο: http://en.wikipedia.org/wiki/HTTP_header_injection
Ανακτήθηκε στις 14 Δεκεμβρίου 2014

75. minsky.gsi.dit.upm.es (2013), HTTP Request Smuggling
Πηγή από το διαδίκτυο:
http://minsky.gsi.dit.upm.es/semanticwiki/index.php/HTTP_Request_Smuggling
Ανακτήθηκε στις 14 Δεκεμβρίου 2014

76. opensourceforu.com (2011), Security Apache, Part 5: HTTP Message Architecture
Πηγή από το διαδίκτυο: http://www.opensourceforu.com/2011/01/securing-apache-part-5-http-message-architecture/
Ανακτήθηκε στις 15 Δεκεμβρίου 2014

77. lists.webappsec.org (2006), Whitepaper by Amit Klein: "HTTP Response Smuggling"

Πηγή από το διαδίκτυο:
http://lists.webappsec.org/pipermail/websecurity_lists.webappsec.org/2006-February/000836.html
Ανακτήθηκε στις 14 Δεκεμβρίου 2014

78. en.wikipedia.org (2014), HTTP response splitting
Πηγή από το διαδίκτυο: http://en.wikipedia.org/wiki/HTTP_response_splitting
Ανακτήθηκε στις 15 Δεκεμβρίου 2014

79. acunetix.com, CSRF Injection Attacks and HTTP Response Splitting
Πηγή από το διαδίκτυο: http://www.acunetix.com/websitesecurity/crlf-injection/
Ανακτήθηκε στις 15 Δεκεμβρίου 2014

80. veracode.com, CSRF Injection Tutorial: Learn About CSRF Injection Vulnerabilities and Prevention
Πηγή από το διαδίκτυο: http://www.veracode.com/security/crlf-injection
Ανακτήθηκε στις 16 Δεκεμβρίου 2014

81. golemtechnologies.com, HTTP Response Splitting
Πηγή από το διαδίκτυο: http://www.golemtechnologies.com/articles/http-response-splitting
Ανακτήθηκε στις 15 Δεκεμβρίου 2014

82. veracode.com, lDAP Injection Guide: Learn How to Detect LDAP Injections and Improve LDAP Security
Πηγή από το διαδίκτυο: https://www.veracode.com/security/ldap-injection
Ανακτήθηκε στις 13 Δεκεμβρίου 2014

83. en.wikipedia.org (2014),
Πηγή από το διαδίκτυο: http://en.wikipedia.org/wiki/Man-in-the-middle_attack
Ανακτήθηκε στις 11 Δεκεμβρίου 2014

84. el.wikipedia.org (2014), Επίθεση Man-in-the-middle
Πηγή από το διαδίκτυο: http://el.wikipedia.org/wiki/Επίθεση_man-in-the-middle
Ανακτήθηκε στις 11 Δεκεμβρίου 2014

85. owasp.org (2014), Man-in-the-middle attack
Πηγή από το διαδίκτυο: https://www.owasp.org/index.php/Man-in-the-middle_attack
Ανακτήθηκε στις 11 Δεκεμβρίου 2014

86. blog.kaspersky.com (2013), What is Man-in-the-middle Attack?
Πηγή από το διαδίκτυο: http://blog.kaspersky.com/man-in-the-middle-attack/
Ανακτήθηκε στις 11 Δεκεμβρίου 2014

87. owasp.org (2009), Man-in-the-browser attack

Πηγή από το διαδίκτυο: https://www.owasp.org/index.php/Man-in-the-browser_attack
Ανακτήθηκε στις 11 Δεκεμβρίου 2014

88. en.wikipedia.org (2014), Man-in-the-browser
Πηγή από το διαδίκτυο: http://en.wikipedia.org/wiki/Man-in-the-browser
Ανακτήθηκε στις 11 Δεκεμβρίου 2014

89. resources.infosecinstitute.com, Man in the browser Attack vs Two Factor Authentication
Πηγή από το διαδίκτυο: http://resources.infosecinstitute.com/two-factor-authentication/
Ανακτήθηκε στις 14 Δεκεμβρίου 2014

90. threatmetrix.com, Man-in-the-browser Detection
Πηγή από το διαδίκτυο: http://www.threatmetrix.com/technology/man-in-the-browser-detection/
Ανακτήθηκε στις 14 Δεκεμβρίου 2014

91. en.wikipedia.org (2014), Browser security
Πηγή από το διαδίκτυο: http://en.wikipedia.org/wiki/Browser_security
Ανακτήθηκε στις 9 Δεκεμβρίου 2014

92. projects.webappsec.org (2010), Null Byte Injection
Πηγή από το διαδίκτυο: http://projects.webappsec.org/w/page/13246949/Null Byte Injection
Ανακτήθηκε στις 17 Δεκεμβρίου 2014

93. hackthis.co.uk (2012), Common PHP Attacks: Poison Null Byte
Πηγή από το διαδίκτυο: https://www.hackthis.co.uk/articles/common-php-attacks-poison-null-byte
Ανακτήθηκε στις 1 Δεκεμβρίου 2014

94. en.wikipedia.org (2014), Directory traversal attack
Πηγή από το διαδίκτυο: http://en.wikipedia.org/wiki/Directory_traversal_attack
Ανακτήθηκε στις 11 Δεκεμβρίου 2014

95. owasp.org (2009), Path Traversal
Πηγή από το διαδίκτυο: https://www.owasp.org/index.php/Path_Traversal
Ανακτήθηκε στις 14 Δεκεμβρίου 2014

96. tinfoilsecurity.com (2014), Path Traversal in Plain English
Πηγή από το διαδίκτυο: https://www.tinfoilsecurity.com/blog/what-is-path-traversal
Ανακτήθηκε στις 21 Δεκεμβρίου 2014

97. acunetix.com, Directory Traversal Attacks
Πηγή από το διαδίκτυο: http://www.acunetix.com/websitesecurity/directory-traversal/
Ανακτήθηκε στις 21 Δεκεμβρίου 2014

98. veracode.com, Directory Traversal
Πηγή από το διαδίκτυο: http://www.veracode.com/security/directory-traversal
Ανακτήθηκε στις 21 Δεκεμβρίου 2014

99. en.wikipedia.org (2014), Phishing
Πηγή από το διαδίκτυο: http://en.wikipedia.org/wiki/Phishing
Ανακτήθηκε στις 21 Δεκεμβρίου 2014

100. identitytheftkiller.com, Are You Phising For Trouble?
Πηγή από το διαδίκτυο: http://www.identitytheftkiller.com/prevent-phishing-scams.php
Ανακτήθηκε στις 21 Δεκεμβρίου 2014

101. blog.infiniteskills.com (2014),
PHP Security: Common Vulnerabilities and Tips to Avoid Attacks
Πηγή από το διαδίκτυο: http://blog.infiniteskills.com/2014/03/php-security-common-vulnerabilities-tips-avoid-attacks/
Ανακτήθηκε στις 21 Δεκεμβρίου 2014

102. Ryan C. Barnett (2006), Preventing Web Attacks With Apache, Pearson Education, Inc

103. infosecpro.com, Predictable Resource Location
Πηγή από το διαδίκτυο: http://www.infosecpro.com/applicationsecurity/a54.htm
Ανακτήθηκε στις 21 Δεκεμβρίου 2014

104. sitepoint.com (2012), Top 10 PHP Security Vulnerabilities
Πηγή από το διαδίκτυο: http://www.sitepoint.com/top-10-php-security-vulnerabilities/
Ανακτήθηκε στις 11 Δεκεμβρίου 2014

105. Mike Harwood (2011),Security Strategies in Web Applications and Social Networking, Jones & Bartlett Learning

106. en.wikipedia.org (2014), Session fixation
Πηγή από το διαδίκτυο: http://en.wikipedia.org/wiki/Session_fixation
Ανακτήθηκε στις 22 Δεκεμβρίου 2014

107. maravis.com (2008), Session fixation attack
Πηγή από το διαδίκτυο: http://www.maravis.com/library/session-fixation-attack/
Ανακτήθηκε στις 23 Δεκεμβρίου 2014

108. owasp.org (2014),Session fixation
Πηγή από το διαδίκτυο: https://www.owasp.org/index.php/Session_fixation
Ανακτήθηκε στις 21 Δεκεμβρίου 2014

109. en.wikipedia.org (2014), Session poisoning
Πηγή από το διαδίκτυο: http://en.wikipedia.org/wiki/Session_poisoning
Ανακτήθηκε στις 23 Δεκεμβρίου 2014

110. imperva.com, Cookie Poisoning
Πηγή από το διαδίκτυο:
http://www.imperva.com/Resources/Glossary?term=cookie_poisoning
Ανακτήθηκε στις 21 Δεκεμβρίου 2014

111. techopedia.com, Cookie Poisoning
Πηγή από το διαδίκτυο: http://www.techopedia.com/definition/16076/cookie-poisoning
Ανακτήθηκε στις 11 Δεκεμβρίου 2014

112. security.radware.com, Cookie Poisoning
Πηγή από το διαδίκτυο: http://security.radware.com/knowledge-center/DDoSPedia/cookie-poisoning/
Ανακτήθηκε στις 1 Δεκεμβρίου 2014

113. openspot.antithesis.gr (2007), SQL Injections (Μέρος 1ο)
Πηγή από το διαδίκτυο: http://openspot.antithesis.gr/archives/33
Ανακτήθηκε στις 1 Δεκεμβρίου 2014

114. en.wikipedia.org (2014), SQL Injections
Πηγή από το διαδίκτυο: http://en.wikipedia.org/wiki/SQL_injection
Ανακτήθηκε στις 1 Δεκεμβρίου 2014

115. owasp.org (2014),Testing for XPath Injection (OTG-INPVAL-010)
Πηγή από το διαδίκτυο:
https://www.owasp.org/index.php/Testing_for_XPath_Injection_(OTG-INPVAL-010)
Ανακτήθηκε στις 21 Δεκεμβρίου 2014

116. cytinus.wordpress.com (2011), XML archives: XML Attacks
Πηγή από το διαδίκτυο: https://cytinus.wordpress.com/tag/xml-attacks/
Ανακτήθηκε στις 11 Δεκεμβρίου 2014

117. hpenterprisesecurity.com, XML External Entity Injection
Πηγή από το διαδίκτυο:
http://www.hpenterprisesecurity.com/vulncat/en/vulncat/dotnet/xxe_injection.html

Ανακτήθηκε στις 21 Δεκεμβρίου 2014

118. clawslab.nds.rub.de, XML Generic Entity Expansion
Πηγή από το διαδίκτυο:
http://clawslab.nds.rub.de/wiki/index.php/XML_Generic_Entity_Expansion
Ανακτήθηκε στις 21 Δεκεμβρίου 2014

119. ws-attacks.org, XML Injection
Πηγή από το διαδίκτυο: http://www.ws-attacks.org/index.php/XML_Injection
Ανακτήθηκε στις 11 Δεκεμβρίου 2014

120. Δημήτριος Γ. Ραπτοδήμος (2009), Διπλωματική Εργασία: Σχεδίαση και υλοποίηση συστήματος ελέγχου πρόσβασης σε βάσεις προσωπικών δεδομένων με χρήση πλαισίου πολιτικής (policy framework)
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υολογιστών
Εθνικό Μετσόβιο Πολυτεχνείο

121. Κατσίδου Μαρία (2007), Μεταπτυχιακή εργασία: Ασφάλεια Ηλεκτρονικών Συναλλαγών και Internet. Η περίπτωση των Εικονικών Εταιριών
Δια τμηματικό Πρόγραμμα Μεταπτυχιακών Σπουδών στα Πληροφοριακά Συστήματα

122. Μιχαήλ Χ. Μεταξάς (2006), Διπλωματική εργασία: Ασφάλεια Ηλεκτρονικών Συναλλαγών στο Διαδίκτυο
Σχολή Χημικών Μηχανικών
Εθνικό Μετσόβιο Πολυτεχνείο

123. islab.demokritos.gr, Secure Socket Layer (SSL)
Πηγή από το διαδίκτυο:
http://www.islab.demokritos.gr/gr/html/ptixiakes/kostas-aris_ptyxiakh/Phtml/ssl.htm
Ανακτήθηκε στις 11 Δεκεμβρίου 2014

124. papaki.gr (2014), Πιστοποιητικά SSL από το Α -Ω!
Πηγή από το διαδίκτυο: https://www.papaki.gr/blog/2014/05/30/πιστοποιητικά-ssl/
Ανακτήθηκε στις 21 Δεκεμβρίου 2014

125. computer.howstuffworks.com, What is a smart card?
Πηγή από το διαδίκτυο: http://computer.howstuffworks.com/question332.htm
Ανακτήθηκε στις 21 Δεκεμβρίου 2014

126. smartcardalliance.org, About Smart Cards
Πηγή από το διαδίκτυο: http://www.smartcardalliance.org/smart-cards-intro-primer/
Ανακτήθηκε στις 21 Δεκεμβρίου 2014

127. square-enix.com, Strengthen your account security with the One-Time Password system!
Πηγή από το διαδίκτυο: http://www.square-enix.com/na/account/otp/
Ανακτήθηκε στις 21 Δεκεμβρίου 2014

128. internetsafetyproject.org, Security Token
Πηγή από το διαδίκτυο: https://www.internetsafetyproject.org/wiki/security-token
Ανακτήθηκε στις 21 Δεκεμβρίου 2014

129. pokerstars.gr, RSA Security Token
Πηγή από το διαδίκτυο:
http://www.pokerstars.gr/poker/room/features/security/rsa-token/faq/
Ανακτήθηκε στις 21 Δεκεμβρίου 2014

130. pcsteps.gr (2013), Τι είναι το VPN - Virtual Private Network - και γιατί μπορεί να χρειάζεστε ένα
Πηγή από το διαδίκτυο: http://www.pcsteps.gr/1376-vpn-technology-explained/
Ανακτήθηκε στις 21 Δεκεμβρίου 2014

131. Ιωάννης Χρυσάνθου (2011), Πτυχιακή Εργασία: Ασφάλεια δικτύων με τεχνικές VPN και SSL,
Τμήμα Επιστήμης και Τεχνολογίας Τηλεπικοινωνιών
Πανεπιστήμιο Πελοποννήσου

132. techworm.net (2015), Top 10 VPN Service providers which ensure that your browsing stays private and anonymous
Πηγή από το διαδίκτυο: http://www.techworm.net/2015/03/top-10-vpn-service-providers-which-ensure-your-browsing-stays-private-anonymous.html
Ανακτήθηκε στις 1 Μάιου 2015

133. web-resources.eu (2011), 22 PHP Frameworks για ταχύτερη ανάπτυξη
Πηγή από το διαδίκτυο: http://www.web-resources.eu/archives/22-php-frameworks-to-shorten-your-development-time
Ανακτήθηκε στις 21 Δεκεμβρίου 2014

134. Ανδρέας Σκουφής & Σωτήρης Σταμουκώστας (2010), Διπλωματική εργασία: Ασφαλής διαμοιρασμός πόρων με εξασφάλιση ανωνυμίας πρόσβασης σε περιβάλλοντα προσωπικών δικτύων
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Εθνικό Μετσόβιο Πολυτεχνείο

135. webapptester.com (2015), Κατασκευή Responsive Ιστοσελίδας με χρήση του Twiter Bootstrap 3.3.1 Μέρος 1ο

Πηγή από το διαδίκτυο: http://webapptester.com/kataskeui-responsive-istoselidas-me-xrisi-tou-responsive-grid-framework/
Ανακτήθηκε στις 2 Μαρτίου 2015

136. elated.com (2011), Build a CMS in an Afternoon with PHP and MySQL
Πηγή από το διαδίκτυο: http://www.elated.com/articles/cms-in-an-afternoon-php-mysql/
Ανακτήθηκε στις 1 Μαρτίου 2014

137. culttt.com (2012), Roll your own PDO PHP Class
Πηγή από το διαδίκτυο: http://culttt.com/2012/10/01/roll-your-own-pdo-php-class/
Ανακτήθηκε στις 1 Μαρτίου 2014

138. jhtmlarea.codeplex.com (2014), jHtmlArea - WYSIWYG HTML Editor for jQuery
Πηγή από το διαδίκτυο: http://jhtmlarea.codeplex.com/
Ανακτήθηκε στις 1 Μαρτίου 2014

139. docs.joomla.org, Search Engine Friendly URLs
Πηγή από το διαδίκτυο: https://docs.joomla.org/Search_Engine_Friendly_URLs
Ανακτήθηκε στις 2 Μαρτίου 2014

140. docs.joomla.org, Preconfigured htaccess
Πηγή από το διαδίκτυο: https://docs.joomla.org/Preconfigured_htaccess
Ανακτήθηκε στις 1 Μαρτίου 2014

141. tophost.gr, SSL Πιστοποιητικά
Πηγή από το διαδίκτυο: http://www.tophost.gr/ssl-certificates.htm
Ανακτήθηκε στις 11 Μαρτίου 2015

142. tinsology.net (2009),Creating a secure login system the right way
Πηγή από το διαδίκτυο: http://tinsology.net/2009/06/creating-a-secure-login-system-the-right-way/
Ανακτήθηκε στις 21 Μαρτίου 2014

143. wikihow.com (2014), Create a Secure Login Script in PHP and MySQL
Πηγή από το διαδίκτυο: http://www.wikihow.com/Create-a-Secure-Login-Script-in-PHP-and-MySQL
Ανακτήθηκε στις 15 Μαρτίου 2014

144. blackbe.lt, Securing PHP User Authentication, Login, And Sessions
Πηγή από το διαδίκτυο: http://blackbe.lt/php-secure-sessions/
Ανακτήθηκε στις 21 Δεκεμβρίου 2014

145. code.tutsplus.com (2014), How to build rate limiting into your-web app login

Πηγή από το διαδίκτυο: http://code.tutsplus.com/tutorials/how-to-build-rate-limiting-into-your-web-app-login--cms-22133
Ανακτήθηκε στις 4 Ιανουαρίου 2015

146. culttt.com (2013), How to save PHP Sessions to a Database
Πηγή από το διαδίκτυο: http://culttt.com/2013/02/04/how-to-save-php-sessions-to-a-database/
Ανακτήθηκε στις 21 Μαρτίου 2014

147. (2015),Create a Secure Session Managment System in PHP and MySQL
Πηγή από το διαδίκτυο: http://www.wikihow.com/Create-a-Secure-Session-Managment-System-in-PHP-and-MySQL
Ανακτήθηκε στις 19 Μαρτίου 2015

148. en.wikibooks.org (2015),
Πηγή από το διαδίκτυο:
http://en.wikibooks.org/wiki/PHP_Programming/Sessions
Ανακτήθηκε στις 21 Μαρτίου 2014

149. sitepoint.com (2011), PHP Sessions
Πηγή από το διαδίκτυο: http://www.sitepoint.com/php-sessions/
Ανακτήθηκε στις 21 Δεκεμβρίου 2014

150. code.google.com, Cool php captcha
Πηγή από το διαδίκτυο: https://code.google.com/p/cool-php-captcha/
Ανακτήθηκε στις 21 Δεκεμβρίου 2014

151. oasp.org, Session Prediction
Πηγή από το διαδίκτυο: https://www.owasp.org/index.php/Session_Prediction
Ανακτήθηκε στις 1 Δεκεμβρίου 2014

152. redteamsecure.com, Demistifying Cross Site Request Forgery
Πηγή από το διαδίκτυο:
http://www.redteamsecure.com/labs/post/66/Demystifying-Cross-Site-Request-Forgery
Ανακτήθηκε στις 1 Δεκεμβρίου 2014

153. facecrooks.com, Beware of Socially Engineered Phishing Attacks on Facebook
Πηγή από το διαδίκτυο: http://facecrooks.com/Scam-Watch/beware-of-socially-engineered-phishing-attacks-on-facebook.html/
Ανακτήθηκε στις 2 Δεκεμβρίου 2014

154. rus-linux.net, HTTP Request Smuggling
Πηγή από το διαδίκτυο: http://rus-linux.net/MyLDP/server/securing-apache-part-5-http-message-architecture.html
Ανακτήθηκε στις 14 Δεκεμβρίου 2014

155. blog.criticalwatch.com, Analysis of the Directory Traversal vulnerability by Basecamp Labs
Πηγή από το διαδίκτυο:
http://blog.criticalwatch.com/post/40788529165/analysis-of-the-directory-traversal-vulnerability
Ανακτήθηκε στις 20 Δεκεμβρίου 2014

156. en.wikipedia.org, Security Token
Πηγή από το διαδίκτυο: https://en.wikipedia.org/?title=Security_token
Ανακτήθηκε στις 20 Δεκεμβρίου 2014