ΧΑΡΟΚΟΠΕΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΤΗΛΕΜΑΤΙΚΗΣ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Αυτοματοποιημένη συλλογή κι αξιολόγηση βασικών πολιτικών ασφάλειας από ιστοσελίδες, με στόχο την εξαγωγή συμπεράσματος για την ασφάλειά τους

Δρακωνάκης Κωνσταντίνος

Επιβλέποντες: **Ριζομυλιώτης Παναγιώτης**, Λέκτορας
Αναγνωστόπουλος Δημοσθένης, Πρύτανης
Καμαλάκης Θωμάς, Λέκτορας

ΑΘΗΝΑ

ΣΕΠΤΕΜΒΡΙΟΣ 2015

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Αυτοματοποιημένη συλλογή κι αξιολόγηση βασικών πολιτικών ασφάλειας από ιστοσελίδες, με στόχο την εξαγωγή συμπεράσματος για την ασφάλειά τους

Δρακωνάκης Κωνσταντίνος
ΑΜ: 21109

ΕΠΙΒΛΕΠΟΝΤΕΣ:

Ριζομυλιώτης Παναγιώτης, Λέκτορας
Αναγνωστόπουλος Δημοσθένης, Πρύτανης
Καμαλάκης Θωμάς, Λέκτορας

Περίληψη

Το διαδίκτυο σήμερα αποτελεί αναπόσπαστο κομμάτι της καθημερινότητας της πλειοψηφίας των πολιτών, ανεξαρτήτως πλέον ηλικίας, και μέρα με την μέρα οι χρήστες του αυξάνονται ραγδαία. Οι συνεισφορές του στην ζωή μας ποικίλουν από διασκέδαση, δικτύωση, ενημέρωση, έως κι εκπόνηση εργασίας είτε επαγγελματικής, είτε ακαδημαϊκής φύσεως. Πέρα από τις ανέσεις και διευκολύνσεις όμως που προσφέρει, το διαδίκτυο ενέχει και διάφορους κινδύνους συσχετισμένους με την ανωνυμία, την προστασία προσωπικών δεδομένων κι άλλα.

Η παρούσα εργασία στοχεύει στην ανάπτυξη μιας εφαρμογής για υπολογιστικά συστήματα, η οποία συλλέγει δεδομένα και χαρακτηριστικά συσχετισμένα με την ασφάλεια από έναν ή περισσότερους ιστότοπους, τα αξιολογεί κι εν συνεχεία εξάγει στατιστικά αποτελέσματα με βάση τα δεδομένα αυτά. Για την ανάδειξη της λειτουργικότητάς της, η εφαρμογή εκτελέστηκε σε ένα σύνολο ελληνικών και μη ιστοσελίδων, που απαρτίζεται από τις κορυφαίες 500 ιστοσελίδες σε επισκεψιμότητα στην Ελλάδα, με απώτερο στόχο την εξαγωγή συμπεράσματος για το κατά πόσο λαμβάνεται υπόψιν η ασφάλεια στο ελληνικό διαδίκτυο και το κατά πόσο η περιήγηση των Ελλήνων χρηστών του διαδικτύου είναι ασφαλής.

Συγκεκριμένα, η εφαρμογή δέχεται ως είσοδο μία ή περισσότερες διευθύνσεις ιστοσελίδων καθώς και κάποιες δευτερεύουσες παραμέτρους εκτέλεσης σύμφωνα με τις οποίες επισκέπτεται τις ιστοσελίδες αυτές και μέσω παθητικής ανάλυσης συλλέγει συγκεκριμένα χαρακτηριστικά από την καθεμία. Στην συνέχεια, αξιολογεί τα χαρακτηριστικά αυτά με βάση μία δεδομένη μετρική και καταγράφει τα αποτελέσματα για κάθε ιστοσελίδα ξεχωριστά. Τέλος, εξάγει στατιστικά αποτελέσματα για το σύνολο των ιστοσελίδων, τα οποία μπορούν εν συνεχεία να αναλυθούν περαιτέρω από το χρήστη για την εξαγωγή συμπερασμάτων που αφορούν τις ιστοσελίδες αυτές.

ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ: Ασφάλεια στο διαδίκτυο, Εφαρμογή Python

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: Παθητική Ανάλυση Ασφάλειας, Αξιολόγηση Ασφάλειας, Crawling, Python, Ελληνικό διαδίκτυο

Abstract

Today, the Internet is an integral part of the majority of the population's daily life, independently from their age, and day by day its users increment dramatically. Its contributions to our lives vary and include, among others, entertainment, networking, information, as well as working, either in a professional or academic manner. Other than the comforts and useful features that it provides, the Internet is subject to numerous hazards that are related to anonymity, personal data protection and more.

This thesis aims at the development of a desktop application for personal computers, which collects data and features related to security by one or more websites, evaluates them and finally produces statistical results based on these data. For the display of its functionality, the application was executed on a set of Greek and foreign websites, which consists of the 500 top visited websites in Greece, with the ultimate purpose of the extraction of a conclusion, concerning the weight that is given to web security in the Greek Internet, as well as if the browsing of the Greek users is considered safe.

Specifically, the application takes as input one or more website addresses, as well as a few secondary arguments according to which it visits the websites. and by using passive analysis it collects specific features from each one of them. Consequently, it evaluates those features based on a certain metric system and records the results for each website individually. Finally, it produces statistical results for the whole set of the visited websites, which can be analyzed later on by the user of the application, for the extraction of further conclusions.

SUBJECT AREA: Web application security, Python application

KEYWORDS: Passive Security Analysis, Security assessment, Crawling, Python, Greek Internet

Ευχαριστίες

Η παρούσα πτυχιακή εργασία διεκπεραιώθηκε με τη συμβολή ορισμένων ατόμων, τόσο του προσωπικού όσο και του ακαδημαϊκού μου κύκλου. Ο καθένας ξεχωριστά με τον δικό του τρόπο, άμεσα κι έμμεσα, με βοήθησε να ολοκληρώσω την πτυχιακή μου εργασία επιτυχώς, καθώς επίσης και τις προπτυχιακές σπουδές μου. Θα ήθελα, λοιπόν, στο σημείο αυτό, να εκφράσω τις θερμότερες ευχαριστίες μου προς τα πρόσωπά τους.

Αρχικά, θα ήθελα να ευχαριστήσω ιδιαιτέρως όλους τους καθηγητές και το προσωπικό του Τμήματος Πληροφορικής & Τηλεματικής του Χαροκοπείου Πανεπιστημίου Αθηνών, για όλες τις γνώσεις και συμβουλές που μου προσέφεραν με υπομονή, επιμονή κι αφοσίωση, προκείμενου να καταφέρω να ολοκληρώσω επιτυχώς τις προπτυχιακές σπουδές μου, αλλά και να έχω το θεμελιώδες ερέθισμα να συνεχίσω να μαθαίνω και να εξελίσσω την επιστήμη μας.

Στην συνέχεια, θα ήθελα να εκφράσω τις ευχαριστίες μου στον επιβλέποντα καθηγητή της πτυχιακής μου εργασίας, κύριο Παναγιώτη Ριζομυλιώτη, για την συνεχή καθοδήγησή του, τις πολύτιμες συμβουλές του, καθώς επίσης και για την άμεση ανταπόκρισή του σε κάθε απορία που συνάντησα καθόλη τη διάρκεια εκπόνησης της πτυχιακής μου εργασίας.

Τέλος, θα ήθελα να ευχαριστήσω από καρδιάς τους κοντινούς μου ανθρώπους, οικογένεια και φίλους, για την υπομονή κι ανοχή που έδειξαν, κυρίως όμως για την ενθάρρυνση και τη στήριξη που μου έδωσαν τόσο κατά την διάρκεια της εκπόνησης της πτυχιακής μου εργασίας, όσο και κατά την διάρκεια των προπτυχιακών μου σπουδών.

Περιεχόμενα

Κατάλογος Σχημάτων

Κατάλογος Πινάκων

Κεφάλαιο 1

Εισαγωγή

Το διαδίκτυο αποτελεί ένα εργαλείο της σύγχρονης καθημερινότητας το οποίο έχει ήδη καταστεί απαραίτητο για τους περισσότερους ανθρώπους. Οπουδήποτε σχεδόν κι αν βρίσκεται κάποιος πλέον έχει εύκολη πρόσβαση στο διαδίκτυο κι επομένως την δυνατότητα να διαχειρίζεται όλη του την δραστηριότητα σε αυτό η οποία συμπεριλαμβάνει μεταξύ άλλων ενημέρωση, κοινωνική δικτύωση, ηλεκτρονική διαχείριση τραπεζικών λογαριασμών, online αγορές, ακόμη κι απομακρυσμένη αποθήκευση προσωπικών αρχείων. Είναι προφανές ότι οι δραστηριότητες αυτές πρέπει να γίνονται με ασφάλεια και να διασφαλίζεται ότι ο χρήστης θα έχει την δυνατότητα να κάνει αυτό που επιθυμεί, χωρίς να υπάρχει κίνδυνος να εκτεθούν ή να τροποποιηθούν προσωπικά κι ευαίσθητα δεδομένα. Το εύρος των απειλών στο διαδίκτυο είναι πολύ μεγάλο και το κόστος αυτών σε βάρος των χρηστών εξίσου πολύ. Δεν είναι δύσκολο άλλωστε να το αναλογιστεί αυτό κανείς, αν σκεφτεί το ενδεχόμενο όπου κάποιος, με κάποιον τρόπο, αποκτά πρόσβαση στα στοιχεία της πιστωτικής κάρτας ενός χρήστη την οποία ο τελευταίος χρησιμοποίησε σε κάποια ιστοσελίδα για μία online αγορά. Από εκεί και πέρα το που θα καταλήξουν οι πληροφορίες αυτές και πώς θα μεταχειριστούν είναι στο χέρι του υποκλοπέα. Καταλαβαίνουμε, λοιπόν, ότι οι υπεύθυνοι των ιστοσελίδων πρέπει να μεριμνούν ώστε τέτοια σενάρια να αποφεύγονται και να μην διακυβεύεται η ασφάλεια του χρήστη και των προσωπικών του δεδομένων κατά την περιήγησή του στην ιστοσελίδα τους.

1.1 Σκοπός της πτυχιακής εργασίας

Ο σκοπός της εργασίας αυτής είναι η ανάπτυξη μιας εφαρμογής που θα συλλέγει δημόσια διαθέσιμα δεδομένα σχετικά με την ασφάλεια από μία ιστοσελίδα και θα μπορεί να τα αξιολογήσει, συμπεραίνοντας

προσεγγιστικά κατά πόσο η ιστοσελίδα αυτή έχει δημιουργηθεί με την μέριμνα της ασφάλειάς της και κατ'επέκταση των χρηστών της. Ακόμη, θα έχει την δυνατότητα να εκτελεί την διαδικασία αυτή μαζικά σε πολλές ιστοσελίδες (crawling) και να παρουσιάζει στατιστικά αποτελέσματα για την περαιτέρω εξαγωγή συμπερασμάτων από τον χρήστη, που θα αφορούν το σύνολο των ιστοσελίδων αυτών.

Συγκεκριμένα, στην παρούσα εργασία η εφαρμογή θα εκτελεστεί μαζικά με στόχο τις 500 κορυφαίες σε επισκεψιμότητα ιστοσελίδες στην Ελλάδα, ώστε να παρουσιαστούν τα ευρήματά της και να σχηματιστεί μία εικόνα για το κατά πόσο οι ελληνικές ιστοσελίδες λαμβάνουν πράγματι την ασφάλεια υπόψιν τους, αλλά ταυτόχρονα και κατά πόσο οι Έλληνες χρήστες του διαδικτύου είναι ασφαλείς κατά την περιήγησή τους στον ιστό.

1.2 Προτεινόμενη Προσέγγιση

Η εφαρμογή έχει γραφτεί εξ ολοκλήρου στην γλώσσα προγραμματισμού Python 2.7.3 κι έχει δοκιμαστεί σε περιβάλλον GNU/Linux. Δέχεται τα ορίσματα του χρήστη από την γραμμή εντολών και στην συνέχεια εκτελεί τις ανάλογες διαδικασίες. Κατά βάση επισκέπτεται τις ιστοσελίδες που έχει ορίσει ο χρήστης, συλλέγει συγκεκριμένες πληροφορίες για αυτές και στην συνέχεια, εφαρμόζοντας μία συγκεκριμένη μετρική, τις αξιολογεί ανάλογα με την βαρύτητα τους, σύμφωνα με το κατά πόσο δηλαδή επηρεάζουν την ασφάλεια της ιστοσελίδας. Εν τέλει, υπολογίζεται ένα σκορ για κάθε ιστοσελίδα και μαζί με τις υπόλοιπες πληροφορίες καταγράφονται σε ένα αρχείο, σε συγκεκριμένη μορφή.

Ο χρήστης μπορεί να ορίσει τις ιστοσελίδες που θέλει να αξιολογηθούν είτε γράφοντας τις διευθύνσεις τους σε ένα αρχείο, είτε τρέχοντας την εφαρμογή σε κάθε μία ξεχωριστά. Μπορεί ακόμη να ορίσει αν η εφαρμογή θα τρέξει σε single mode ή σε crawl mode, αν δηλαδή θα αξιολογηθεί η ιστοσελίδα που ορίζεται μόνο ή αν θα αξιολογηθούν και οι ιστοσελίδες που περιέχονται ως σύνδεσμοι στην αρχική ιστοσελίδα.

Στην περίπτωση του crawl mode, ο χρήστης έχει και την δυνατότητα να ορίσει το βάθος του crawling. Έτσι, δημιουργείται ένα δέντρο και το crawling, και κατ'επέκταση και η αξιολόγηση, ακολουθούν την προσέγγιση ενός DFS (Depth First Search) αλγορίθμου.

Οι πληροφορίες για κάθε ιστοσελίδα συλλέγονται μέσω παθητικής ανάλυσης (passive analysis) και χωρίζονται σε δύο κατηγορίες: HTTP response headers και HTML response. Στην πρώτη κατηγορία ελέγχεται η ύπαρξη και η τιμή συγκεκριμένων HTTP headers, τα οποία σχετίζονται με την ασφάλειά, που επιστρέφονται από την ιστοσελίδα, όπως αυτά ορίζονται από το Open Web Application Security Project (OWASP) [1]. Στην δεύτερη κατηγορία ελέγχεται η ύπαρξη συγκεκριμένων στοιχείων στον HTML κώδικα της ιστοσελίδας.

Η μετρική που χρησιμοποιείται βασίζεται στο Common Vulnerability Scoring System Version 3.0

(CVSS v3.0) [2]. Η μετρική αυτή είναι προσεγγιστική, αφού κάθε στοιχείο που ελέγχεται δεν πρόκειται, κατά κανόνα, για μία ευπάθεια (vulnerability), αλλά για έναν μηχανισμό ασφάλειας (defense mechanism) που αποσκοπεί στην προστασία της ιστοσελίδας από μία συγκεκριμένη ευπάθεια. Επομένως, σε κάθε τέτοιο μηχανισμό έχει ανατεθεί ένα σκορ που θα αντιστοιχούσε στην ευπάθεια την οποία αποτρέπει.

Για την καταγραφή των αποτελεσμάτων χρησιμοποιείται μορφοποίηση JSON, αλλά και η φυσική γώσσα, και η αποθήκευση γίνεται σε τοπικά αρχεία. Με αυτόν τον τρόπο μπορούν να προσπελαστούν εύκολα τα αποτελέσματα της εκτέλεσης μετέπειτα και να εξαχθούν τα στατιστικά αποτελέσματα, όπως επίσης δίνεται η δυνατότητα στον χρήστη να μπορεί να ανατρέξει σε αυτά για να τα μελετήσει περαιτέρω οποιαδήποτε στιγμή.

1.3 Ασφάλεια

1.3.1 Βασικές έννοιες

Η ασφάλεια των πληροφοριών ως έννοια και ως ανάγκη προέκυψε από τις πρώτες μορφές επικοινωνίας. Η ανάγκη για προστασία ευαίσθητων πληροφοριών, όπως π.χ. διπλωματικών, κυβερνητικών και στρατιωτικών, ώθησε τους ανθρώπους στο να βρουν τρόπους ώστε να την διασφαλίσουν. Με το πέρασμα των αιώνων ο τρόπος με τον οποίο διασφαλιζόταν η εμπιστευτικότητα και ακεραιότητα των πληροφοριών εξελισσόταν κι ενώ στην αρχή βασιζόταν αποκλειστικά στην φυσική προστασία τους, αργότερα βρέθηκαν νέοι τρόποι ώστε ακόμα κι αν οι πληροφορίες υποκλέπτονταν, να μην ήταν άμεσα εφικτό για τον υποκλοπέα να τις ερμηνεύσει. Σημαντικό σημείο στην ιστορία της ασφάλειας των πληροφοριών που υποδεικνύει την εξέλιξη των μέσων προστασίας τους, είναι η δημιουργία του κρυπτογραφήματος του Καίσαρα (Caesar's cipher), περίπου το 50 π.Χ. [3]

Από τα μέσα του 20ου αιώνα μέχρι και σήμερα, μια εποχή που σηματοδοτήθηκε από την ραγδαία κι ευρεία χρήση διασυνδεδεμένων ηλεκτρονικών υπολογιστών, η έννοια της ασφάλειας των πληροφοριών απέκτησε βαρύτητα μεγαλύτερη από ποτέ. Αυτό γιατί πλέον μέσω του διαδικτύου οι πληροφορίες ανταλλάσσονται συνεχώς και η φυσική μόνο προστασία των τερματικών που αποθηκεύουν τις πληροφορίες δεν αρκεί.

Σήμερα, έχει καθιερωθεί η ασφάλεια των πληροφοριών να χαρακτηρίζεται από τρεις βασικές αρχές, την εμπιστευτικότητα, την ακεραιότητα και την διαθεσιμότητα.

1.3.1.1 Εμπιστευτικότητα

Η εμπιστευτικότητα (confidentiality) είναι η ιδιότητα που ορίζει ότι οι πληροφορίες δεν γίνονται διαθέσιμες ή αποκαλύπτονται σε μη εξουσιοδοτημένα άτομα, οντότητες ή διεργασίες. [4]

1.3.1.2 Ακεραιότητα

Η ακεραιότητα (integrity) ορίζεται ως η ιδιότητα που διασφαλίζει ότι η πληροφορία δεν θα τροποποιηθεί μη εξουσιοδοτημένα ή χωρίς τον εντοπισμό της τροποποίησης, καθ'όλη την διάρκεια του κύκλου ζωής της. [4]

1.3.1.3 Διαθεσιμότητα

Η διαθεσιμότητα (availability) είναι η ιδιότητα που διασφαλίζει ότι η πληροφορία θα είναι διαθέσιμη όταν χρειαστεί. Αυτό σημαίνει ότι οποιοσδήποτε μηχανισμός ασφάλειας χρησιμοποιείται (πρωτόκολλα, διεργασίες, υλικό) πρέπει να λειτουργεί σωστά, καθώς επίσης και να αντιμετωπίζονται απειλές που μπορεί να διακόψουν την ορθή λειτουργία του συστήματος. [4]

1.3.2 Ασφάλεια στο διαδίκτυο

Η ασφάλεια στο διαδίκτυο αναφέρεται κατά κύριο λόγο στην ασφάλεια των εφαρμογών του, είτε πρόκειται για απλές ιστοσελίδες, είτε για δυναμικές, είτε για υπηρεσίες και οι περισσότερες απειλές οφείλονται σε προγραμματιστικά λάθη στην υλοποίηση της εκάστοτε εφαρμογής.

1.3.2.1 Απειλές

Οι απειλές στις διαδικτυακές εφαρμογές διαχωρίζονται σε 6 κατηγορίες, καλύπτουν διαφορετικές γενικές περιπτώσεις και η καθεμία επηρεάζει έναν μηχανισμό ασφάλειας, όπως παρουσιάζεται παρακάτω.

- Πλαστογραφία (Spoofing): Απόκρυψη της ταυτότητάς κάποιου και προσποίηση κάποιου άλλου

προσώπου.
Μηχανισμός που επηρεάζεται: Πιστοποίηση (Authentication)

- Αλλοίωση (Tampering): Κακόβουλη και μη εξουσιοδοτημένη τροποποίηση δεδομένων.
  Μηχανισμός που επηρεάζεται: Ακεραιότητα (Integrity)
- Αποκήρυξη (Repudiation): Διεκπεραίωση απαγορευμένων ενεργειών σε κάποιο σύστημα που δεν μπορεί να εντοπίσει τις ενέργειες αυτές.
  Μηχανισμός που επηρεάζεται: Μη-αποκήρυξη (Non-repudiation)
- Αποκάλυψη πληροφοριών (Information Disclosure): Η αποκάλυψη πληροφοριών σε κάποιον που κανονικά δεν θα είχε πρόσβαση σε αυτές.
  Μηχανισμός που επηρεάζεται: Εμπιστευτικότητα (Confidentiality)
- Άρνηση εξυπηρέτησης (Denial of Service): Άρνηση εξυπηρέτησης σε έγκυρους χρήστες, καθιστώντας το σύστημα ή την υπηρεσία μη διαθέσιμη ή χρησιμοποιήσιμη.
  Μηχανισμός που επηρεάζεται: Διαθεσιμότητα (Availability)
- Ανύψωση δικαιωμάτων (Elevation of Privilege): Απόκτηση προνομιούχας πρόσβασης σε πόρους για την μη εξουσιοδοτημένη πρόσβαση σε πληροφορίες ή για την κατάληψη του συστήματος.
  Μηχανισμός που επηρεάζεται: Εξουσιοδότηση (Authorization) [5]

1.3.2.2 Αληθινά περιστατικά

Για να κατανοήσει κανείς την σημαντικότητα της ασφάλειας στα πληροφοριακά συστήματα, αρκεί να αναλογιστεί τα παρακάτω αληθινά περιστατικά που επηρέασαν εκατομμύρια ανθρώπους. Φυσικά, έχουν υπάρξει πολύ περισσότερες περιπτώσεις, αλλά αυτές που ακολουθούν ξεχωρίζουν για τον αντίκτυπό τους.

Heartland Payment Systems, 2008-2009

Στην προκειμένη περίπτωση, ένα κακόβουλο λογισμικό στο σύστημα της εταιρείας κατέγραφε δεδομένα πιστωτικών καρτών από το δίκτυο της, κάτι που εν τέλει προκάλεσε την διαρροή περίπου 130 εκ. πιστωτικών καρτών, την μεγαλύτερη μέχρι στιγμής διαρροή δεδομένων στις Η.Π.Α. Η εν λόγω εταιρία πληρωμών είναι από τις μεγαλύτερες στις Η.Π.Α. και διαχειριζόταν τις πληρωμές για πάνω από 250,000 επιχειρήσεις, πράγμα στο οποίο οφείλεται και ο τεράστιος αντίκτυπος της παραβίασης της ασφάλειάς της.[6]

Sony Online Entertainment Services, 2011

Σε αυτό το περιστατικό, εισβολείς διείσδυσαν στο PlayStation Network της Sony που διασυνδέει οικιακές κονσόλες βιντεοπαιχνιδιών και διέρρευσαν προσωπικά στοιχεία, όπως ονοματεπώνυμα, τηλέφωνα και διευθύνσεις, 79 εκ. χρηστών. Στην συνέχεια διαπιστώθηκε όμως ότι οι εισβολείς είχαν διεισδύσει και σε κάποιες άλλες υπηρεσίες και ο συνολικός αριθμός των επηρεασμένων χρηστών έφτασε τα 102 εκ., αυτήν την φορά με στοιχεία πιστωτικών καρτών ανάμεσα στα δεδομένα που υποκλάπηκαν. Το περιστατικό όμως δεν επηρέασε μόνο τους χρήστες των οποίων προσωπικά δεδομένα αποκαλύφθηκαν, αλλά και την ίδια την Sony η οποία αναγκάστηκε να κλείσει εντελώς το Playstation Network για πάνω από 3 εβδομάδες, αλλά και να αντιμετωπίσει 65 αγωγές εναντίον της, ανυψώνοντας έτσι την ζημιά της εταιρίας στα 171 εκατομμύρια δολάρια. [6]

1.4 Web-Crawling και Python

1.4.1 Web-Crawling

Το web-crawling είναι η διαδικασία αυτοματοποιημένης προσπέλασης πολλών ιστοσελίδων, ξεκινώντας από μία αρχική διεύθυνση. Συνήθως, ο σκοπός του είναι η ευρετηρίαση των ιστοσελίδων και αποτελεί βασικό συστατικό στοιχείο των μηχανών αναζήτησης. Συγκεκριμένα, οι crawlers των μηχανών αναζήτησης επισκέπτονται, αποθηκεύουν κι ευρετηριάζουν ιστοσελίδες έτσι ώστε η αναζήτηση των χρηστών να γίνεται πολύ πιο αποδοτικά.

Παρά ταύτα, υπάρχουν και crawlers που στόχος τους δεν είναι η αποθήκευση κι ευρετηρίαση αλλά η συλλογή κι εξαγωγή δεδομένων και γνώσης από ιστοσελίδες, όπως αυτός που κατασκευάστηκε για την παρούσα εργασία.

1.4.1.1 Web Crawling VS Web Scraping

Το web crawling και το web scraping είναι δύο έννοιες συσχετισμένες, συχνά αλληλοσυμπληρωματικές, αρκετά όμοιες, αλλά και με πολύ συγκεκριμένες διαφορές. Το web crawling στοχεύει βασικά στην αποθήκευση και ευρετηρίαση σελίδων με την χρήση bots ή crawlers. Βασικό του χαρακτηριστικό όμως παραμένει η μαζική επίσκεψη μεγάλου πλήθους ιστοσελίδων και η ανάκτηση και επίσκεψη νέων διευθύνσεων κι έχει καθιερωθεί να χαρακτηρίζεται κι από αυτό. Το web scraping στοχεύει

στην συλλογή πολύ συγκεκριμένων πληροφοριών και πιθανώς από συγκεκριμένες ιστοσελίδες και άρα μπορεί να υπάρξει ανεξάρτητα από το web crawling, π.χ. η εξαγωγή δεδομένων από μία συγκεκριμένη ιστοσελίδα.

Επομένως, είναι φανερή η αλληλεξάρτηση των δύο τεχνικών. Το web crawling, προκειμένου να ανακτήσει νέους συνδέσμους για να συνεχίσει την εκτέλεσή του, πρέπει να τους εξάγει από την κάθε σελίδα που επισκέπτεται, άρα να εφαρμόσει μία μορφή web scraping. Από την άλλη, το web scraping, μπορεί να σταθεί μόνο του, αλλά στην περίπτωση που θέλουμε να εξάγουμε συγκεκριμένη πληροφορία από ένα μεγάλο εύρος σελίδων, όπως στην παρούσα εργασία, πρέπει να ενσωματωθεί σε έναν web crawler.

1.4.1.2 Πολιτική crawling

Η συμπεριφορά και τρόπος δράσης ενός crawler καθορίζεται από ένα σύνολο πολιτικών, οι οποίες διαμορφώνονται ανάλογα με τον σκοπό του:

- Πολιτική επιλογής (selection) – Ποιες ιστοσελίδες πρέπει να κατεβάσει ο crawler
- Πολιτική επανεπισκεψιμότητας (re-visit) – Πότε να γίνει έλεγχος για αλλαγές στις σελίδες
- Πολιτική “ευγένειας” (politeness) – Πώς να αποφευχθεί η υπερφόρτωση ενός ιστοτόπου
- Πολιτική παραλληλισμού (parallelization) – Πώς να συγχρονίζονται κατανεμημένοι web crawlers[7]

1.4.1.3 Αναγνωριστικά των crawlers

Οι web crawlers συνήθως παρέχουν κάποιο αναγνωριστικό της ταυτότητάς τους στέλνοντας μαζί με κάθε αίτηση σε μία σελίδα το πεδίο User-agent. Έτσι οι διαχειριστές των σελίδων διαβάζοντας τα αρχεία καταγραφής (logs) της ιστοσελίδας τους μπορούν να δουν πότε και πόσο συχνά την επισκέπτεται κάποιος crawler. Ακόμη, σε περίπτωση που κάποιος crawler πέσει σε μία “παγίδα” (crawler trap), αν στο αναγνωριστικό του περιέχεται κάποια διεύθυνση επικοινωνίας με τον ιδιοκτήτη του, τότε ο διαχειριστής της σελίδας μπορεί να επικοινωνήσει μαζί του ώστε να τον σταματήσει. [8]

1.4.1.4 Ζητήματα ασφάλειας

Αν και οι περισσότεροι ιδιοκτήτες ιστοσελίδων θέλουν οι σελίδες τους να αποτελούν σημεία επίσκεψης των διάφορων crawlers, ώστε να έχουν καλύτερη κατάταξη στα αποτελέσματα μηχανών αναζήτησης, το crawling μίας σελίδας μπορεί, ακούσια, να έχει κάποιες παρενέργειες που να πλήξουν την ασφάλεια του ιστότοπου.

Το πιο πιθανό ζήτημα που μπορεί να προκύψει είναι η ευρετηρίαση σελίδων που δεν θα έπρεπε να είναι διαθέσιμες δημόσια, όπως back-up αρχεία βάσεων δεδομένων ή ακόμη και αρχεία με κωδικούς χρηστών ή άλλες ευαίσθητες πληροφορίες. Με αυτόν τον τρόπο, επιτιθέμενοι μπορούν με συγκεκριμένα ερωτήματα στην μηχανή αναζήτησης να ψάξουν για τέτοια αρχεία και να βρουν ευάλωτα σημεία του στόχου τους. Υπάρχει βέβαια τρόπος αποτροπής των μηχανών αναζήτησης να ευρετηριάζουν μη δημόσια κομμάτια ιστοσελίδων κι αυτός είναι να ορίζεται το αρχείο “robots.txt” στον αρχικό φάκελο κάθε ιστοσελίδας, το οποίο ορίζει ουσιαστικά ποιες σελίδες επιτρέπεται να επισκεφτεί ένας crawler. [9]

1.4.2 Python

Η Python είναι μία γλώσσα προγραμματισμού υψηλού επιπέδου και γενικού σκοπού που επιτρέπει την πιο γρήγορη εργασία και την αποδοτικότερη υλοποίηση συστημάτων. [10] Στην επίσημη ιστοσελίδα της Python αναφέρεται:

“Python is powerful... and fast;
plays well with others;
runs everywhere;
is friendly & easy to learn;
is Open” [11]

1.4.2.1 Βασικά χαρακτηριστικά

Multi-paradigm

Βασικό χαρακτηριστικό της Python, είναι η υποστήριξη πολλών διαφορετικών προγραμματιστικών παραδειγμάτων, όπως αντικειμενοστραφή (object-oriented), δομημένο/διαδικαστικό (structured/procedural) και συναρτησιακό (functional) προγραμματισμό.

Dynamic typing

Η Python κάνει χρήση δυναμικού ελέγχου τύπων, δηλαδή ελέγχει την συνάφεια των τύπων συναρτήσεων ή μεταβλητών κατά την εκτέλεση του προγράμματος. Έτσι, δεν χρειάζεται ο προγραμματιστής να δηλώνει αποκλειστικά τον τύπο μιας μεταβλητής.

Memory Management

Η Python συνδυάζοντας ένα σύστημα μέτρησης των αναφορών σε πόρους (reference counting) με έναν "συλλέκτη σκουπιδιών" (garbage collector), μπορεί να κάνει αποδοτική διαχείριση της μνήμης και να αποδεσμεύει πόρους που δεν χρησιμοποιούνται πλέον.

Dynamic name resolution (late binding)

Ενώ άλλες γλώσσες αντιστοιχίζουν τα ονόματα και τους τύπους των συναρτήσεων κατά το compilation, η Python το κάνει κατά την εκτέλεση. Αν κληθεί μία συνάρτηση, για παράδειγμα., τότε θα ψάξει εκείνη την ώρα με βάση το όνομα της αν υπάρχει και στην συνέχεια θα την καλέσει.

Semantics and simple syntax

Η Python χαρακτηρίζεται από το απλό συντακτικό της και τους κανόνες που το διέπουν, κάτι που στοχεύει στην ευκολότερη εκμάθηση της γλώσσας και την απλούστερη συγγραφή κώδικα.

1.4.2.2 Ιστορία κι εξέλιξη

Η υλοποίηση της Python ξεκίνησε στην Ολλανδία το 1989, από τον Guido van Rossum και βασίστηκε σε μία άλλη γλώσσα προγραμματισμού, την ABC. Σκοπός του Rossum ήταν να κρατήσει κάποια από τα καλά χαρακτηριστικά της ABC και να αφαιρέσει κάποια άλλα. Ο κύριος λόγος για την δημιουργία της γλώσσας ήταν να μπορεί να διαχειρίζεται σφάλματα (exception handling) αλλά και να είναι κατάλληλη για αλληλεπίδραση με το λειτουργικό σύστημα Amoeba.

Η πρώτη έκδοση της γλώσσας (0.9.0) δημοσιεύτηκε τον Φεβρουάριο του 1991 από τον Rossum. Από αυτήν την έκδοση ακόμα υπήρχαν κλάσεις και κληρονομικότητα, διαχείριση σφαλμάτων, συναρτήσεις και βασικοί τύποι δεδομένων. Από την πρώτη της μορφή, η Python φαίνεται ότι έχει επηρεαστεί από άλλες γλώσσες προγραμματισμού, όπως από την Modula-3 για το σύστημα ενοτήτων της (module system) και την διαχείριση σφαλμάτων. Το 1994 σχηματίζεται το πρώτο forum συζήτησης για την Python, το comp.lang.python, κάτι που αποτέλεσε ορόσημο για το ευρύ κοινό φίλων της γλώσσας.

Ακολούθησαν μερικές εκδόσεις ακόμη με σημαντικότερες την έκδοση 2.0 στις 16 Οκτωβρίου 2000

και την 3.0 στις 3 Δεκεμβρίου του 2008. Στην έκδοση 2.0 ξαναπαρουσιάζεται η επιρροή της Python από άλλες γλώσσες με την νέα λειτουργικότητα της “κατανόησης λίστας” (list comprehension), η οποία ήταν γνωστή από τις γλώσσες SETL και Haskell. Ακόμη, σε αυτήν την έκδοση εμφανίστηκε και το σύστημα “συλλογής σκουπιδιών” (garbage collection) που αποτελεί βασικό χαρακτηριστικό της γλώσσας. Στις εκδόσεις της ίδιας σειράς που ακολούθησαν ξεχωρίζει η 2.2, όπου συνενώθηκαν οι τύποι της Python (γραμμένοι σε C) και οι κλάσεις της (γραμμένες σε Python) σε μία κοινή ιεραρχία, κάτι που κατέστησε το μοντέλο της γλώσσας ξεκάθαρα αντικειμενοστραφή. Στην ίδια έκδοση προστέθηκαν και οι γεννήτριες (generators), εμπνευσμένες από την γλώσσα Icon.

Η έκδοση 3.0 σχεδιάστηκε για να διορθώσει ορισμένα σχεδιαστικά ελαττώματα της γλώσσας και ο λόγος που απέκτησε καινούργιο όνομα σειράς (η σειρά 2 σταμάτησε στην έκδοση 2.7) ήταν ότι δεν ήταν εφικτή η συμβατότητα με τις προηγούμενες εκδόσεις. Στόχος της νέας σειράς ήταν να μειώσει τον αριθμό των ενοτήτων (modules) που είχαν σχεδιαστεί για την εκπλήρωση της ίδιας εργασίας, με βασική αρχή το “Θα έπρεπε να υπάρχει ένας – και κατά προτίμηση μόνο ένας – προφανής τρόπος να το κάνεις”. Παρά ταύτα και αυτή η έκδοση διατήρησε την ύπαρξη πολλών προγραμματιστικών παραδειγμάτων.

Σήμερα επικρατούν δύο βασικές εκδόσεις της Python, η 2.7.10 και η 3.4.3. και με τις δύο να είναι ευρέως διαδεδομένες, όμως με την 3.4 να έχει πιο βέβαιο μέλλον κι εξέλιξη, αφού η 2.7 θα υποστηρίζεται μέχρι και το 2020. Αυτό γιατί αναμένεται οι χρήστες να μεταβούν στις εκδόσεις 3.4+ το συντομότερο. [12]

1.5 Δομή της εργασίας

Η παρούσα πτυχιακή εργασία απαρτίζεται από 6 κεφάλαια. Κάθε ένα από αυτά εστιάζει σε ένα διαφορετικό συστατικό της στοιχείο και το αναλύει, επεξηγώντας βασικές έννοιες και στοιχεία που η εργασία πραγματεύεται.

Αρχικά, παρουσιάζονται διάφορες εισαγωγικές πληροφορίες, όπως ο σκοπός της πτυχιακής εργασίας, καθώς και η προτεινόμενη προσέγγιση για την υλοποίηση της εφαρμογής. Ακόμη, παρατίθενται κι επεξηγούνται βασικές έννοιες της ασφάλειας των πληροφοριών, αλλά και των πληροφοριακών συστημάτων, καθώς επίσης και έννοιες που σχετίζονται με το Web-Crawling. Τέλος, περιγράφονται τα βασικά χαρακτηριστικά της γλώσσα προγραμματισμού Python, όπως και μία σύντομη ιστορική αναδρομή της.

Εν συνεχεία, στο 2ο κεφάλαιο, αναλύεται το θεωρητικό υπόβαθρο της πτυχιακής εργασίας, επεξηγώντας λεπτομερώς τις ευπάθειες κι επιθέσεις με τις οποίες ασχολείται, όπως επίσης και τους μηχανισμούς άμυνας που ελέγχει. Ακόμη, γίνεται σαφής η διαφορά της ενεργητικής και παθητικής ανάλυσης και ποια προσέγγιση ακολουθεί η εργασία. Τέλος, περιγράφεται η μετρική που διαμορφώθηκε και χρησιμοποιήθηκε για την αξιολόγηση των ιστοσελίδων.

Στο 3ο κεφάλαιο, δίνεται τόσο μία λεκτική, όσο και οπτική περιγραφή της αρχιτεκτονικής της

εφαρμογής, με εστίαση στο crawl mode εκτέλεσης. Επίσης, περιγράφονται τα βασικά εργαλεία της γλώσσας προγραμματισμού Python που χρησιμοποιήθηκαν για την υλοποίηση των διάφορων λειτουργιών της εφαρμογής.

Στο 4ο κεφάλαιο, περιγράφεται κι επεξηγείται λεπτομερώς η υλοποίηση της εφαρμογής, οι διάφορες κλάσεις και οι αρμοδιότητές τους, όπως επίσης και βασικές λειτουργίες της εφαρμογής, π.χ. η σύνδεση με τις ιστοσελίδες, η συλλογή κι αξιολόγηση των διάφορων στοιχείων και η λειτουργικότητα του crawling.

Το επόμενο κεφάλαιο, παρουσιάζει ένα σενάριο χρήσης της εφαρμογής και συγκεκριμένα την εκτέλεσή της σε ένα σύνολο ιστοσελίδων. Περιγράφονται τα χαρακτηριστικά του συνόλου και της εκτέλεσης και τα ευρήματά της πάνω στο σύνολο αυτό.

Στο 6ο και τελευταίο κεφάλαιο, αναφέρονται τα συμπεράσματα μετά το πέρας της εκπόνησης της εργασίας, που αφορούν τα διάφορα ζητήματα και περιορισμούς που έπρεπε να ληφθούν υπόψιν, αλλά και το κατά πόσο η εργασία ανταποκρίθηκε εν τέλει στους αρχικούς της στόχους. Τέλος, περιγράφονται ορισμένες πιθανές επεκτάσεις και βελτιώσεις που μπορούν να γίνουν εν καιρώ, στην εφαρμογή που αναπτύχθηκε.

Κεφάλαιο 2

Θεωρητικό Υπόβαθρο

2.1 Ευπάθειες, επιθέσεις και μηχανισμοί άμυνας

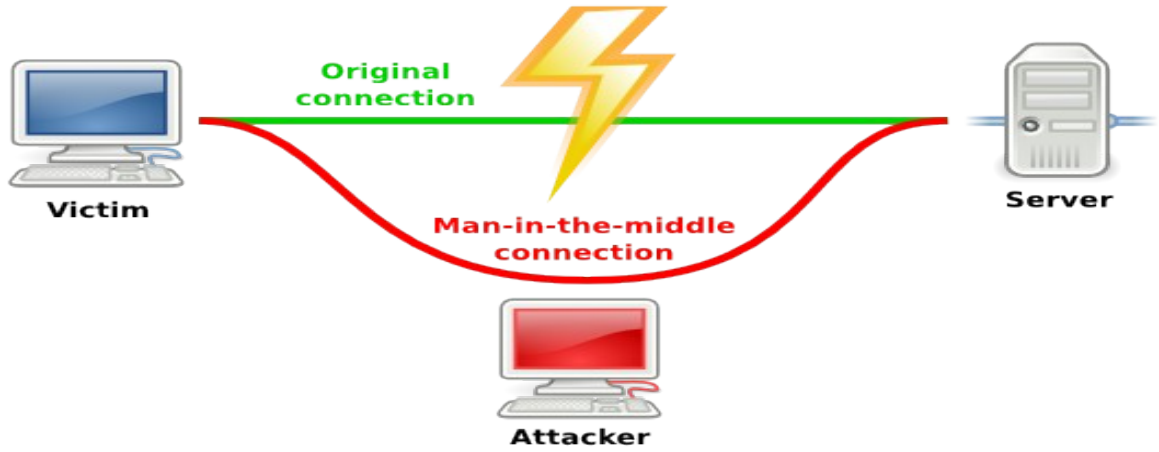
Όπως προαναφέρθηκε στην παράγραφο 1.3.2 η ασφάλεια στο διαδίκτυο αφορά τις εφαρμογές και υπηρεσίες που εκτελούνται σε αυτό και φτάνουν στον χρήστη μέσω του φυλλομετρητή (browser) του. Οι απειλές που υπάρχουν έρχονται σε πέρας με την εκμετάλλευση κάποιων συγκεκριμένων ευπαθειών (vulnerabilities). Συγκεκριμένα, ο παράγοντας στον οποίο οφείλονται οι περισσότερες, αν όχι όλες, ευπάθειες κι επομένως απειλές δεν είναι άλλος από τον ανθρώπινο. Οφείλονται δηλαδή σε προγραμματιστικά λάθη και συνήθως στην παραμέληση ελέγχου της εισόδου του χρήστη (user-input sanitization).

2.1.1 Ευπάθειες κι επιθέσεις

Οι ευπάθειες που έχουν βρεθεί κι αναγνωριστεί επίσημα είναι πάρα πολλές. Μάλιστα, συνεχώς ανακαλύπτονται νέες, είτε σε νέα, είτε σε παλιότερα λογισμικά (0-day exploits). Η παρούσα εργασία πραγματεύεται 5 συγκεκριμένες από τις πιο διαδεδομένες ευπάθειες κι επιθέσεις. Αξίζει να σημειωθεί ότι δεν γίνεται προσπάθεια εκμετάλλευσής τους, γιατί αυτό θα παραβίαζε την ισχύουσα νομοθεσία, παρά ταύτα παρατηρείται η ύπαρξη μηχανισμών για την αποτροπή τους.

2.1.1.1 Man in The Middle Attack (MiTM)

Η επίθεση man in the middle, αποτελεί μία από τις παλιότερες επιθέσεις και βασικό της χαρακτηριστικό είναι η παρακολούθηση της συνομιλίας μεταξύ δύο υπολογιστών, πιθανώς ενός server κι ενός client. Η παρακολούθηση μπορεί να είναι είτε ενεργή είτε παθητική, δηλαδή ο ενδιάμεσος και πιθανώς κακόβουλος χρήστης μπορεί να προσπαθεί να τροποποιήσει τα δεδομένα που ανταλλάσσονται μεταξύ των δύο άλλων μελών ή απλά να τα “ακούει” και να τα καταγράφει.



Σχήμα 2.1: Σχηματική Αναπαράσταση μιας MiTM επίθεσης *Πηγή:tails.boum.org*

Βέβαια, σε κάθε περίπτωση ο επιτιθέμενος δεν θέλει να αποκαλύψει την παρουσία του, για τον λόγο αυτό προσποιείται, ανάλογα με τον ποιόν μιλάει εκείνη την στιγμή, το άλλο μέλος της συνομιλίας.

Ένα απλοποιημένο παράδειγμα μιας τέτοιας επίθεσης είναι το σενάριο που ακολουθεί. Ο ένας χρήστης, π.χ. Alice, θέλει να επικοινωνήσει με κάποιον άλλον, π.χ. τον Bob. Επομένως, του στέλνει ένα μήνυμα που του λέει ποια είναι, χωρίς όμως αυτό να εξακριβώνεται κάπως, και του ζητάει το δημόσιό του κλειδί έτσι ώστε να αρχίσει η κρυπτογραφημένη τους συνομιλία. Στο ενδιάμεσο όμως, βρίσκεται ο επιτιθέμενος, π.χ. Eve. Αν καταφέρει να “ακούσει” το μήνυμα της Alice, μπορεί να το προωθήσει εκείνη στον Bob προσποιούμενη την Alice. Ο Bob δεν μπορεί να πει με βεβαιότητα ότι δεν μιλάει με την Alice, οπότε απαντάει στο μήνυμα με το δημόσιο του κλειδί. Και πάλι με την σειρά της η Eve συλλαμβάνει το μήνυμα του Bob, το αποτρέπει από το να φτάσει στην Alice και αντί αυτού της στέλνει το δικό της δημόσιο κλειδί. Στην συνέχεια η Alice κρυπτογραφεί με το κλειδί της Eve, νομίζοντας ότι μιλάει στον Bob και η Eve διαβάζει όλα τα μηνύματα κανονικά και μπορεί ακόμα και να τα τροποποιήσει. Μπορεί να σταματήσει την συνομιλία με τον Bob, ή μπορεί να του προωθεί τα μηνύματα της Alice ώστε να μην γίνει αντιληπτή. Η Eve κάνει τις αντίστοιχες ενέργειες κι όταν επικοινωνεί με τον Bob, προσποιούμενη την Alice.

Στην εργασία αυτή, εξετάζεται μία συγκεκριμένη ευπάθεια που μπορεί να την εκμεταλλευτεί κάποιος επιτιθέμενος μέσω μιας MiTM επίθεσης ώστε να την επεκτείνει σε επόμενες συνδέσεις του χρήστη με τον server. Η ευπάθεια είναι γνωστή ως SSL-Stripping κι εντοπίζεται σε ευάλωτες HTML φόρμες. Αν η σύνδεση σε μία σελίδα έχει γίνει μέσω του πρωτοκόλλου HTTP (μη κρυπτογραφημένη συνομιλία) και σε αυτήν την σελίδα υπάρχει μία φόρμα που δείχνει σε κάποια σελίδα με το πρωτόκολλο HTTPS (κρυπτογραφημένη συνομιλία), τότε ο επιτιθέμενος μέσω μιας MiTM επίθεσης μπορεί να αλλάξει τις ενέργειες των φορμών αυτών ώστε να χρησιμοποιούν HTTP. Έτσι καταφέρνει να εφαρμόσει την επίθεση και σε επόμενες συνδέσεις, που υπό κανονικές συνθήκες θα ήταν ασφαλείς.

2.1.1.2 Cross-Site Scripting (XSS)

Η επίθεση αυτή είναι από τις πιο διαδεδομένες και χαρακτηρίζεται από την δυνατότητα κάποιου να εισάγει εκτελέσιμο από τον browser κώδικα (JavaScript, HTML, Flash κλπ) σε μία σελίδα. Αποτελεί ένα πολύ καλό παράδειγμα για το πόσο σημαντικός είναι ο έλεγχος και καθαρισμός των δεδομένων εισόδου του χρήστη. Η επίθεση χωρίζεται σε δύο βασικές κατηγορίες:

Stored/Persistent XSS

Στην περίπτωση αυτή ο κώδικας που μπήκε στην σελίδα παραμένει εκεί μόνιμα και εκτελείται στον browser κάθε χρήστη που επισκέπτεται την σελίδα. Ο αντίκτυπός του είναι αρκετά σημαντικότερος από την δεύτερη κατηγορία που θα περιγραφεί παρακάτω, γιατί η επίθεση δεν είναι στοχευμένη σε μεμονωμένους χρήστες, αλλά στο σύνολό των επισκεπτών της σελίδας. Για παράδειγμα, έστω ότι ένας ιστότοπος έχει μία κοινότητα με forum. Αν δεν γίνεται σωστός έλεγχος των μηνυμάτων που αναρτούν οι χρήστες τότε κάποιος μπορεί πολύ εύκολα να αναρτήσει ένα μήνυμα που αντί για κανονικό κείμενο να περιέχει κώδικα JavaScript, π.χ. :

<script>document.location='evilsite.com/cookie_stealer.php?cookie='+document.cookie</script>

Αν ο κακόβουλος χρήστης καταφέρει να αναρτήσει αυτόν τον κώδικα στην σελίδα, τότε κάθε χρήστης που την επισκέπτεται θα ανακατευθύνεται σε μία εξωτερική σελίδα και μαζί θα δίνονται και ως παράμετροι τα cookies του χρήστη που ανήκουν στον ευάλωτο ιστότοπο. Ο επιτιθέμενος, στην συνέχεια, έχοντας αποθηκεύσει τις πληροφορίες μπορεί να τις χρησιμοποιήσει για να μπει στην ιστοσελίδα σαν κάποιος άλλος χρήστης.

Reflected/Non-persistent XSS

Αντίθετα με την πρώτη περίπτωση, εδώ ο κώδικας δεν παραμένει στην σελίδα αλλά η εκτέλεσή του προκύπτει από την εμφάνιση κάποιου μηνύματος λάθους ή από την κατασκευή κάποιου κακόβουλου

συνδέσμου. Η επίθεση εδώ συνήθως γίνεται στοχευμένα σε μεμονωμένους χρήστες, ξεγελώντας τους ώστε να ακολουθήσουν κάποιον σύνδεσμο κατασκευασμένο έτσι ώστε με το που μεταφερθεί εκεί ο χρήστης να εκτελεστεί ο κώδικας. Για παράδειγμα, έστω ότι μια σελίδα, η index.php, σε κάποιον ιστότοπο έχει μία φόρμα, η οποία στον HTML κώδικα φαίνεται κάπως έτσι:

<form name=”my_form” action=”<?php echo $_SERVER['PHP_SELF'] ?>”>

Παρατηρούμε, ότι περιέχεται κώδικας PHP στην ενέργεια της φόρμας και συγκεκριμένα λέει στην φόρμα να στείλει τα δεδομένα της στην ίδια σελίδα. Αν τοποθετηθεί στην διεύθυνση της σελίδας το εξής και κάποιος ανοίξει τον σύνδεσμο, με την προϋπόθεση πάντα ότι τα δεδομένα εισόδου δεν ελέγχονται:

http://www.example.site.com/index.php/”><script src=”evil.site.com/evil_script.js”/>

Τότε ο παραπάνω HTML κώδικας θα γίνει:

<form name=”my_form” action=”index.php/”><script src=”evil.site.com/evil_script.js”/>>

Επομένως, θα φορτωθεί το εξωτερικό κακόβουλο script και θα εκτελεστεί στον browser του χρήστη.

2.1.1.3 Cross Site Request Forgery (CSRF)

Αυτή η επίθεση βασίζεται στο να ξεγελαστεί ο χρήστης και να πατήσει κάποιον σύνδεσμο ή να υποβάλλει μία φόρμα, που η πραγματική τους λειτουργία δεν είναι φανερή, αλλά συγκαλυμμένη από κάποια άλλη φαινομενικά αθώα ενέργεια. Αξίζει να σημειωθεί, ότι εδώ ο επιτιθέμενος δεν στοχεύει στο να υποκλέψει πληροφορίες, διότι ο κανονικός χρήστης είναι εκείνος που θα υποβάλει την φόρμα και θα πάρει πίσω την απάντηση. Αντίθετα, στοχεύει στην αλλαγή κάποιας κατάστασης στον ιστότοπο που στοχεύει ο κακόβουλος σύνδεσμος ή φόρμα.

Για παράδειγμα, ο επιτιθέμενος γνωρίζει ότι το θύμα του είναι πελάτης μίας συγκεκριμένης τράπεζας και άρα πολύ πιθανόν να είναι κι εγγεγραμμένος στο online σύστημά της. Ο επιτιθέμενος έχει μελετήσει το σύστημα συναλλαγών της τράπεζας κι έχει παρατηρήσει ότι μία αίτηση για μία συναλλαγή είναι της μορφής:

https://www.fakebank.com/transact.php?from=AccountOwnerName&to=recipient&amount=10000

Μία προσέγγιση είναι ο επιτιθέμενος να παραπλανήσει τον χρήστη στο να επισκεφτεί την κακόβουλη σελίδα του και να υποβάλλει μία φόρμα της μορφής:

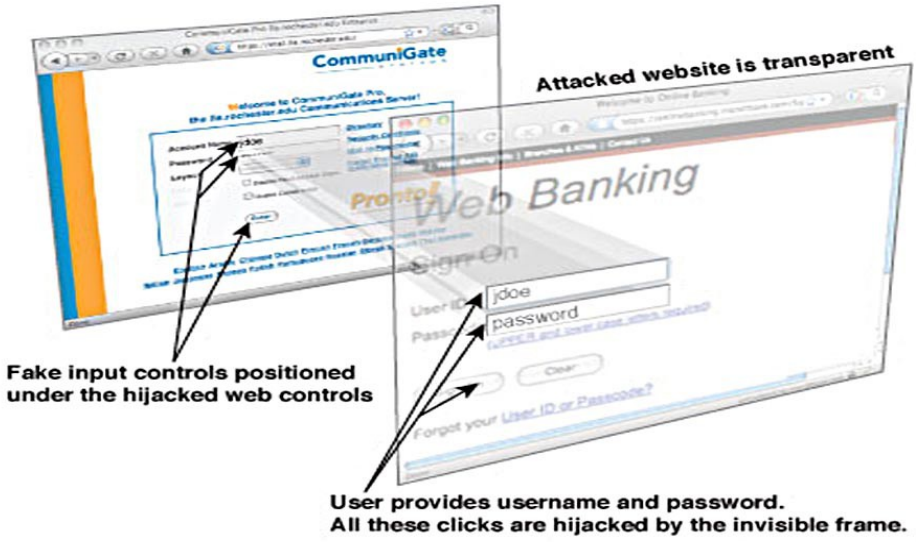
```
<form method=”GET” action=”https://www.fakebank.com/transact.php”>
        Your name: <input name=”name” type=”text”/>
        Your email: <input name=”email” type=”text”/>
        <input type=”hidden” name=”from” value=”VictimAccountName”/>
        <input type=”hidden” name=”to” value=”AttackersAccountName”/>
        <input type=”hidden” name=”amount” value=”10000”/>
</form>
```

Όταν ο χρήστης υποβάλλει την φόρμα και αν δεν έχει προσέξει τα κρυμμένα πεδία, η αίτηση της συναλλαγής θα φτάσει στην σελίδα της τράπεζας. Ο χρήστης δεν χρειάζεται να είναι εκείνη την ώρα στον ιστότοπο της τράπεζας, αρκεί να έχει κρατήσει τα δεδομένα αυθεντικοποίησης που έχει αποθηκεύσει η σελίδα στον υπολογιστή του (cookies). Τέλος, αφού ο χρήστης είναι αυθεντικοποιημένος στην τράπεζα και η σελίδα της τράπεζας δεν μπορεί να ξεχωρίσει αν πρόκειται για μία εσκεμμένη ή ακούσια συναλλαγή, η συναλλαγή θα ολοκληρωθεί.

2.1.1.4 Clickjacking

Η επίθεση αυτή μοιάζει ως έναν βαθμό με το CSRF που περιγράφηκε παραπάνω, με την έννοια ότι ο χρήστης, άθελά του πάλι,προβαίνει σε μια ενέργεια που είναι συγκαλυμμένη από κάποια άλλη. Συγκεκριμένα, ο επιτιθέμενος κατασκευάζει μια ιστοσελίδα με πολλά επίπεδα (layers) όπου έχει στοιχίσει προσεκτικά τα στοιχεία (κουμπιά, συνδέσμους κλπ) της δικής του σελίδας, τα οποία μπορεί να περιέχουν παραπλανητικά μηνύματα, με αυτά της σελίδας στο κορυφαίο επίπεδο η οποία καταρχήν δεν φαίνεται και κατά δεύτερον είναι μία σελίδα που περιέχει την λειτουργία που θέλει να φέρει εις πέρας ο επιτιθέμενος.

Ένα πιθανό σενάριο είναι ο επιτιθέμενος να πείσει με κάποιον τρόπο το θύμα του να επισκεφτεί την σελίδα του, που μπορεί να περιέχει οποιοδήποτε περιεχόμενο, π.χ. συμμετοχή σε κληρώσεις με δελεαστικά έπαθλα. Στο κορυφαίο επίπεδο, όμως, ο επιτιθέμενος έχει ενσωματώσει μία άλλη σελίδα, ίσως με την χρήση κάποιου <iframe> στοιχείου, χωρίς να φαίνεται, π.χ. κάποια σελίδα κοινωνικής δικτύωσης κι έχει στοιχίσει ένα κουμπί που λέει “Κερδίστε ένα smartphone” έτσι ώστε να είναι ακριβώς πάνω από ένα άλλο της αυθεντικής σελίδας που διαγράφει τις επαφές του χρήστη. Ο χρήστης νομίζοντας ότι πατάει το πρώτο κουμπί, χωρίς να το συνειδητοποιήσει πατάει το δεύτερο κι έτσι η επίθεση πετυχαίνει.



Σχήμα 2.2 Παράδειγμα δομής μιας επίθεσης Clickjacking *Πηγή: 4.bp.blogspot.com*

Όπως φαίνεται και στην εικόνα, οι επιθέσεις αυτού του τύπου δεν στοχεύουν μόνο στην υποκλοπή “clicks”, αλλά και δεδομένων που πληκτρολογεί ο χρήστης σε μία φόρμα της σελίδας.

2.1.1.5 Stale JavaScript Inclusions

Αυτή η συγκεκριμένη περίπτωση πρόκειται για μία ευπάθεια κι όχι άμεσα για μια επίθεση. Ουσιαστικά, έχει να κάνει με την αμέλεια των υπεύθυνων των ιστοσελίδων να ελέγχουν και να διατηρούν τα εξωτερικά περιεχόμενα των ιστοσελίδων τους και πιο συγκεκριμένα την ενσωμάτωση JavaScript κώδικα από τρίτους παρόχους. Ο ένας κίνδυνος που προκύπτει είναι να μην πληκτρολογηθεί σωστά η διεύθυνση του παρόχου ή του εξωτερικού αρχείου, δηλαδή αν η σωστή διεύθυνση είναι:
http://www.fake.site.com/http/random.js
να γραφτεί:
http://www.fake.**stie**.com/http/random.js

Ο δεύτερος κίνδυνος είναι να πάψει να υπάρχει ο πάροχος. Και στις δύο περιπτώσεις ο επιτιθέμενος μπορεί να δηλώσει το λανθασμένο ή μη υπάρχον πλέον domain όνομα και δημιουργώντας τα ίδια ονομαστικά αρχεία, στα ίδια μονοπάτια, να αρχίσει να παρέχει αυτός πλέον τον δικό του κακόβουλο JavaScript κώδικα.. [13]

2.1.2 Μηχανισμοί άμυνας

Προκειμένου να αποφευχθούν και να προληφθούν έως ένα σημείο ευπάθειες κι επιθέσεις, όπως αυτές που περιγράφηκαν στην παράγραφο 2.1.1, έχουν καθιερωθεί κάποιοι μηχανισμοί ασφάλειας ως πρότυπα και ανεξάρτητοι από την υλοποίηση μιας εφαρμογής. Πιο συγκεκριμένα πρόκειται κυρίως για οδηγίες που παίρνει ο browser του χρήστη από τον server που συνδέεται για το πώς να διαχειριστεί ορισμένα δεδομένα κι ενέργειες. Η πλειοψηφία αυτών των μηχανισμών συμπεριλαμβάνεται στις κεφαλίδες HTTP (HTTP headers) που στέλνει ο server στον browser του χρήστη και κάποιοι άλλοι στον HTML κώδικα της σελίδας που επιστρέφεται.

2.1.2.1 Cookies

Τα cookies είναι πληροφορία που αποθηκεύει ο απομακρυσμένος server στον υπολογιστή του χρήστη κι αυτός τα στέλνει πίσω με κάθε νέα του αίτηση. Τα cookies μπορεί να αποθηκεύονται για διάφορους σκοπούς, αλλά συνήθως πρόκειται για την αυθεντικοποίησή του χρήστη και την συνέχιση της συνεδρίας του με τον server. Όπως φαίνεται στην παράγραφο 2.1.1.2, κάποιος επιτιθέμενος μπορεί μέσω μίας επίθεσης Cross-site scripting να υποκλέψει τα cookies κάποιου νόμιμου χρήστη και στην συνέχεια να τον υποδυθεί.

Http-only cookies

Προκειμένου να αποφευχθεί η υποκλοπή των cookies, ο server στην κεφαλίδα που στέλνει ως απάντηση στην αίτηση του χρήστη και την πρώτη φορά που δημιουργεί cookies, μπορεί να ορίσει την ιδιότητα “http-only”. Το πεδίο της κεφαλίδας θα είναι κάπως έτσι:

Set-cookie: PHPSESSID=123456789asdf; Expires=Thu, 25 Aug 2016 21:00:00 GMT; path=/; httponly;

Αυτή η ιδιότητα εξασφαλίζει ότι τα cookies δεν μπορούν να διαβαστούν ή τροποποιηθούν από κώδικα που τρέχει στον υπολογιστή του χρήστη (client-side). Έτσι ακόμη κι αν ο χρήστης πατήσει σε έναν σύνδεσμο που ενεργοποιεί μία XSS επίθεση, ο κώδικας του επιτιθέμενου δεν θα έχει πρόσβαση στα cookies του χρήστη.

Secure cookies

Ακόμη μία ιδιότητα που μπορεί να ενεργοποιηθεί για την προστασία των cookies είναι η “secure”. Το πεδίο

της κεφαλίδας θα είναι κάπως έτσι:

Set-cookie: PHPSESSID=123456789asdf; Expires=Thu, 25 Aug 2016 21:00:00 GMT; path=/; secure;

Η συγκεκριμένη ιδιότητα, εξασφαλίζει ότι τα cookies που την έχουν θα μεταδίδονται μόνο όταν έχει εγκαθιδρυθεί μία ασφαλής συνεδρία μεταξύ χρήστη και server, μέσω του HTTPS πρωτοκόλλου. Άρα άν υπάρχει κάποιος ενδιάμεσος στην συνομιλία και βρίσκεται εν μέσω μιας MiTM επίθεσης, ακόμη κι αν “ακούει” τα δεδομένα που ανταλλάσσουν οι δύο συνομιλούντες τα cookies που περιέχουν ευαίσθητη πληροφορία θα μεταδίδονται είτε κρυπτογραφημένα μέσω HTTPS, είτε καθόλου μέσω HTTP.

2.1.2.2 HTTP Strict-Transport-Security

Το Strict-Transport-Security είναι ένα ακόμα Header field που υποχρεώνει τον browser του χρήστη να χρησιμοποιεί μόνο το HTTPS πρωτόκολλο για την σύνδεση του με τον server, για ένα συγκεκριμένο χρονικό διάστημα. Το πεδίο είναι ως εξής:

Strict-Transport-Security: max-age=31536000; includeSubDomains;

Ακόμη κι αν ο χρήστης προσπαθήσει να συνδεθεί μέσω HTTP, ακολουθώντας π.χ. κάποιον σύνδεσμο, ο browser θα μετατρέψει το HTTP σε HTTPS, πριν συνδεθεί με τον server. Επίσης, αν υπάρχει κάποιο πρόβλημα με την εγκαθίδρυση ασφαλούς σύνδεσης με τον server, π.χ. το πιστοποιητικό του δεν είναι έγκυρο ή έχει λήξει, τότε ένα μήνυμα λάθους θα εμφανιστεί στον χρήστη και δεν θα επιτραπεί καθόλου η σύνδεση.

Συμπερασματικά, ακόμη κι αν κάποιος εκτελέσει μία MiTM επίθεση, είτε ενεργητική είτε παθητική, θα του είναι πολύ δύσκολο να εξάγει πληροφορία και να την ερμηνεύσει κι ακόμη περισσότερο να την τροποποιήσει, αφού τα δεδομένα θα είναι κρυπτογραφημένα.

2.1.2.3 X-Frame-Options

Αυτό το header field ουσιαστικά λέει με ποιόν τρόπο πρέπει ο browser να διαχειριστεί την φόρτωση ενός <iframe> ή <frame> HTML στοιχείου. Τα στοιχεία αυτά επιτρέπουν την ενσωμάτωση άλλων σελίδων σε κάποια άλλη. Πιθανές τιμές για αυτό το πεδίο είναι οι εξής:

- **DENY**, όπου δεν επιτρέπεται σε κανέναν να ενσωματώσει την σελίδα αυτή,
- **SAMEORIGIN**, που επιτρέπει την ενσωμάτωση της σελίδας μόνο από σελίδες του ίδιου ιστοτόπου,

- **ALLOW-FROM *uri***, που επιτρέπει την ενσωμάτωση της σελίδας από συγκεκριμένους ιστοτόπους.

Η επίθεση που αντιμετωπίζεται με τον μηχανισμό αυτό είναι η επίθεση Clickjacking, όπως περιγράφηκε στην παράγραφο 2.1.1.4. Αν κάποιος επιτιθέμενος προσπαθήσει να φορτώσει μία σελίδα στην δική του παραπλανητική και κακόβουλη σελίδα, με την χρήση κάποιου HTML iframe, η ενέργεια αυτή θα αποτραπεί από τον browser του, ο οποίος όταν προσπαθήσει να την φορτώσει θα δει το X-Frame-Options πεδίο και αν δεν ορίζει την σελίδα του επιτιθέμενου ως έγκυρη, τότε θα απαγορέψει την ενέργεια.

2.1.2.4 X-XSS-Protection

Μέσω αυτού του μηχανισμού ενεργοποιείται ή απενεργοποιείται το XSS-Filter, το οποίο είναι ενεργοποιημένο κατά κανόνα σε ορισμένους browsers (Internet Explorer 8+ και τελευταίες εκδόσεις Google Chrome) και στοχεύει στην αποτροπή επιθέσεων Cross-Site Scripting. Ουσιαστικά, το φίλτρο αυτό ελέγχει τις παραμέτρους που στέλνει ο χρήστης στην αίτησή του για μία σελίδα και αν παρατηρήσει κάποια ύποπτη παράμετρο, που θα μπορούσε να είναι π.χ. JavaScript κώδικας, τότε ελέγχει και την επιστρεφόμενη σελίδα. Αν στην απάντηση υπάρχει αυτός ο κώδικας τότε ο browser αποτρέπει την εκτέλεσή του και είτε ειδοποιεί τον χρήστη με κάποιο μήνυμα, είτε σταματάει τελείως την φόρτωση της σελίδας. Η ενεργοποίηση του φίλτρου στον browser γίνεται με το εξής πεδίο κεφαλίδας:

X-XSS-Protection: 1: mode=block

,όπου το mode ορίζει αν θα φορτωθεί η σελίδα ή θα διακοπεί τελείως η φόρτωσή της.

2.1.2.5 X-Content-Type-Options

Αυτό το header field είναι υπεύθυνο για την αποτροπή εκμετάλλευσης ευπαθειών κατά την διαδικασία του mime sniffing που γίνεται από ορισμένους browsers για την αναγνώριση του τύπου των δεδομένων που λαμβάνονται. Ο λόγος που κάποιοι browsers προβαίνουν στην διαδικασία του mime ή content sniffing είναι ότι τα δεδομένα που λαμβάνονται μπορεί να μην έχουν επαρκείς πληροφορίες στα μεταδεδομένα τους ή το πεδίο “Content-type”, που ακριβώς ορίζει τον τύπο των δεδομένων που λαμβάνονται,να μην έχει οριστεί σωστά. Άρα πρέπει να υπάρξει μια άλλη μέθοδος για την αναγνώριση του τύπου τους κι επομένως την κατάλληλη διαχείρισή κι ερμηνεία τους.

Το mime sniffing όμως εισάγει κι μία πιθανή προσέγγιση επίθεσης. Έστω ότι μία ιστοσελίδα δίνει την δυνατότητα στου χρήστες της να ανεβάσουν μία εικόνα και ο έλεγχος που γίνεται είναι να συγκριθεί η

κατάληξη του αρχείου με γνωστές καταλήξεις αρχείων εικόνων (π.χ. .jpg, .png, .gif κλπ). Ο επιτιθέμενος, μπορεί να ανεβάσει ένα αρχείο με όνομα *example_img.jpg* και μέσα σε αυτό να έχει HTML ή PHP κώδικα. Το ψεύτικο αρχείο εικόνας περνάει τον έλεγχο, αφού πληροί την συνθήκη να έχει μία έγκυρη κατάληξη κι εν τέλει ο server μέσω του mime sniffing, καταλαβαίνει ότι πρόκειται για ένα αρχείο PHP και το ερμηνεύει αναλόγως. Αυτό δίνει την δυνατότητα στον επιτιθέμενο να εκτελέσει αυθαίρετο κώδικα στον server, αλλά και στον browser ενός χρήστη που θα επισκεφτεί το αρχείο που ανέβασε.

Το πεδίο X-Content-Type-Options παίρνει μόνο την τιμή "*nosniff*" και δίνει την οδηγία στον browser να μην προβεί σε mime sniffing. Με τον τρόπο αυτό, ακόμη κι αν το παραπάνω πλαστό αρχείο περιέχει κώδικα του επιτιθέμενου, ο browser θα προσπαθήσει να το ανοίξει ως μία JPG εικόνα κι αφού δεν θα είναι ουσιαστικά μία, απλά θα εμφανίσει ένα μήνυμα λάθους στον χρήστη.

2.1.2.6 X-Content-Security-Policy

Πρόκειται για μία πολιτική που ορίζει από που και/ή ποιοι πόροι μπορούν να φορτωθούν στην ιστοσελίδα που έχει αυτό το header field. Έχει συγκεκριμένες οδηγίες και σε συγκεκριμένη μορφοποίηση, μερικές από τις οποίες είναι οι εξής:

- script-src: Από που επιτρέπεται να φορτώνονται scripts
- img-src: Από που επιτρέπεται να φορτώνονται εικόνες
- frame-src: Από που επιτρέπεται να φορτώνονται frames
- form-action: Ποιες διευθύνσεις μπορούν να χρησιμοποιηθούν ως ενέργειες HTML φορμών
- report-uri: Σε ποια διεύθυνση να στέλνονται αναφορές όταν υπάρχει κάποια παραβίαση της πολιτικής [14]

Επομένως πρόκειται για έναν γενικότερο μηχανισμό ασφάλειας που μπορεί να καλύψει διάφορες πτυχές της περιμέτρου ενός ιστοτόπου και να μειώσει την πιθανότητα εκμετάλλευσης κάποιας ευπάθειας.

2.1.2.7 Ιδιότητα sanbox σε iframes

Η ιδιότητα αυτή εμπεριέχεται σε ετικέτες iframe στον HTML κώδικα μίας σελίδας και ορίζει τον τρόπο με τον οποίο θα μεταχειρίζεται και τι ενέργειες θα επιτρέπονται στην ενσωματωμένη σελίδα. Μπορεί να πάρει κάποιες συγκεκριμένες τιμές, όπως:

- allow-forms: Επιτρέπει την υποβολή φορμών από την ενσωματωμένη σελίδα

- allow-pointer-block: Επιτρέπει την χρήση APIs
- allow-popups: Επιτρέπει την εμφάνιση παραθύρων popup
- allow-same-origin: Επιτρέπει την μεταχείριση της ενσωματωμένης σελίδας, σαν να ανήκε στον ίδιο τον ιστότοπο
- allow-scripts: Επιτρέπει την εκτέλεση scripts

Αν δεν δοθεί καμία τιμή στην ιδιότητα αυτή, τότε απαγορεύει όλα τα παραπάνω και προφανώς παρέχει την μεγαλύτερη ασφάλεια σε ό,τι αφορά τα φορτωμένα frames. Αν για παράδειγμα ένας επιτιθέμενος καταφέρει να φορτώσει την δική του ή μία κακόβουλη σελίδα στο iframe της αξιόπιστης σελίδας, ενώ θα μπορούσε να εκτελεί δικό του JavaScript κώδικα στον browser του χρήστη, με την χρήση του *sandbox*, δεν μπορεί και οι δυνατότητές του περιορίζονται κατά πολύ.

2.1.2.8 CSRF tokens

Ένας τρόπος για την αποφυγή μίας επίθεσης Cross-Site Request Forgery (CSRF), όπως αυτή περιγράφηκε στην παράγραφο 2.1.1.3, είναι η χρήστη CSRF tokens που αποστέλλονται μαζί με την υποβολή μίας φόρμας. Πρόκειται για μία κρυμμένη, μεγάλη και δύσκολο στα να υπολογιστεί ή να την μαντέψει κάποιος πληροφορία που γνωρίζει μόνο ο έγκυρος ιστότοπος και συσχετίζεται με την συνεδρία ενός χρήστη. Αν υποβληθεί κάποια φόρμα από τον χρήστη ο ιστότοπος είναι σε θέση να καταλάβει αν αυτή ήταν μία εσκεμμένη ενέργεια του χρήστη ή όχι, ελέγχοντας αν υπάρχει αυτή η πληροφορία κι αν ταιριάζει με αυτήν που έχει συσχετιστεί με τον συγκεκριμένο χρήστη. Ένα πεδίο CSRF token που αποστέλλεται μαζί με τα υπόλοιπα στοιχεία της φόρμας είναι της μορφής:

<input type=”hidden” name=”_token” value=”ed936ca02f1dc2bc8644a126253760c6”/>

2.2 Ενεργητική και παθητική ανάλυση κι επιθέσεις

Η ανάλυση των μηχανισμών ασφάλειας συστημάτων και των δικτύων τους χωρίζεται σε δύο ευρείες κατηγορίες, εξίσου σημαντικές και αλληλοσυμπληρωματικές: την παθητική και την ενεργητική ανάλυση κι επιθέσεις. Κάθε μία έχει τον δικό της σκοπό και διαφορετικό αντίκτυπο στο εκάστοτε στοχευμένο σύστημα, καθώς επίσης και διαφορετική αντιμετώπιση από την ισχύουσα νομοθεσία.

2.2.1 Παθητική ανάλυση κι επιθέσεις

Η παθητική ανάλυση στοχεύει στην συλλογή πληροφοριών για το σύστημα υπό μελέτη και δεν πρόκειται ακριβώς για κάποια επίθεση, αφού ο επιτιθέμενος ή αναλυτής απλά "ακούει" ή συλλέγει πληροφορίες που είναι διαθέσιμα δημόσιες ή απλώς απροστάτευτες. Εν συνεχεία, τα αποτελέσματα της ανάλυσης μπορούν να χρησιμοποιηθούν και να υποδείξουν τεχνικές και προσεγγίσεις για ενεργές επιθέσεις. Αυτού του είδους η ανάλυση, συνήθως συμπεριλαμβάνει την εξέταση πακέτων που αποστέλλονται και λαμβάνονται από το σύστημα που εξετάζεται, για την εξαγωγή ευαίσθητης πληροφορίας όπως αναγνωριστικά χρηστών, κωδικούς, κλειδιά κλπ. Αυτό, όμως, απαιτεί το άτομο που κάνει την ανάλυση να βρίσκεται μέσα στο δίκτυο του στόχου του και ακόμη, αν δεν πρόκειται για δικό του δίκτυο, να έχει την αντίστοιχη, επίσημη άδεια για ανάλυση του δικτύου από τον κάτοχό του, ειδάλλως πρόκειται για μία παράνομη πράξη. Εκτός από αυτό, η παθητική ανάλυση ενός συστήματος, συμπεριλαμβάνει και την εξέταση άλλων δημόσια διαθέσιμων πληροφοριών, όπως τον HTML κώδικα σελίδων και τα HTTP Headers που αποστέλλει ο server. Αυτά είναι που εξετάζει και η παρούσα εργασία και πρόκειται για πληροφορίες που δεν επηρεάζουν το υπό ανάλυση σύστημα, καθώς επίσης δεν παραβιάζουν την σχετική νομοθεσία, μπορούν όμως να υποδείξουν πιθανές ευπάθειες του συστήματος.

2.2.2 Ενεργητική ανάλυση κι επιθέσεις

Αυτού του είδους η ανάλυση στοχεύει στην εκμετάλλευση ευπαθειών, που πιθανώς να αναγνωρίστηκαν κατά την φάση της παθητικής ανάλυσης, και κατά κανόνα έχουν κάποιον αντίκτυπο στο στοχευμένο σύστημα. Μπορεί ακόμα ο αναλυτής να προσπαθεί να βρει ευπάθειες δοκιμάζοντας διάφορες ενεργητικές τεχνικές, όπως π.χ. με το να προσπαθεί να εκτελέσει μια απλή αρχικά επίθεση Cross-Site Scripting (XSS) σε κάποια φόρμα που πιστεύει ότι μπορεί να είναι ευάλωτη. Όπως και με την εξέταση πακέτων, έτσι κι εδώ ο αναλυτής οφείλει να έχει λάβει την αντίστοιχη άδεια από τον κάτοχο του συστήματος, διότι σε οποιαδήποτε άλλη περίπτωση δεν πρόκειται για νόμιμη πράξη.

Αν και ο συγκεκριμένος τύπος ανάλυσης μπορεί να δώσει μία πιο αντικειμενική εικόνα για την ασφάλεια ενός συστήματος, η παρούσα εργασία δεν πραγματοποιεί κάποια ενεργητική επίθεση αφού στοχεύει στην ανάλυση πολλών διαφορετικών ιστοσελίδων και η κατοχύρωση της αντίστοιχης άδειας από τους ιδιοκτήτες τους θα ήταν ανέφικτη.

2.3 Μηχανισμός αξιολόγησης

Ο μηχανισμός αξιολόγησης που χρησιμοποιήθηκε στην εργασία βασίζεται στο Common Vulnerability Scoring System Version 3.0 (CVSS v 3.0) που αναπτύχθηκε από το FIRST (Forum of Incident Response and Security Teams) [2]. Το συγκεκριμένο σύστημα στοχεύει στην αξιολόγηση ευπαθειών και όχι μηχανισμών ασφάλειας. Για τον λόγο αυτό, σε κάθε μηχανισμό ασφάλειας αποδόθηκε ένας βαθμός που αντιστοιχεί στην ευπάθεια που ο μηχανισμός στοχεύει να αποτρέψει.

Συγκεκριμένα, το CVSS v 3.0 αποδίδει τρία διαφορετικά σκορ σε κάθε ευπάθεια, τα οποία είναι:

- **Base score**: Περιγράφει τα γενικότερα χαρακτηριστικά μιας ευπάθειας , που παραμένουν σταθερά ανά την πάροδο του χρόνου και ανά τα διάφορα περιβάλλοντα χρηστών. Τα στοιχεία που πρέπει να συμπληρωθούν για να υπολογιστεί το σκορ είναι**:**
  - **Attack Vector:** Το μέσο από το οποίο ο επιτιθέμενος μπορεί να εκμεταλλευτεί την ευπάθεια (π.χ. δίκτυο, φυσική πρόσβαση).
  - **Attack Complexity:** Περιγράφει την πολυπλοκότητα της επίθεσης και κάποιες άλλες παραμέτρους, όπως την αναγκαία παρουσία ενεργών ρυθμίσεων στον στόχο.
  - **Privileges Required:** Τα δικαιώματα που πρέπει να έχει ο επιτιθέμενος στο στοχευμένο σύστημα προκειμένου να καταφέρει να εκμεταλλευτεί την ευπάθεια..
  - **User Interaction:** Η ανάγκη για κάποια ενέργεια ενός χρήστη, εκτός του επιτιθέμενου, προκειμένου να ολοκληρωθεί με επιτυχία η επίθεση.
  - **Scope:** Περιγράφει την επιρροή κάποιου άλλου στοιχείου του συστήματος, εκτός του ευάλωτου, μετά την επίθεση.
  - **Confidentiality:** Κατά πόσο επηρεάζεται η εμπιστευτικότητα πληροφοριών και πόρων στο σύστημα μετά την επίθεση
  - **Integrity:** Κατά πόσο επηρεάζεται η ακεραιότητα των πληροφοριών και δεδομένων στο σύστημα μετά την επίθεση
  - **Availability:** Κατά πόσο επηρεάζεται η διαθεσιμότητα του ευάλωτου στοιχείου, αλλά και γενικότερα του συστήματος μετά την επίθεση

- **Temporal Score:** Περιγράφει την ύπαρξη τεχνικών και κώδικα για την εκμετάλλευση της ευπάθειας, την ύπαρξη διορθώσεων και μηχανισμών αποτροπής της και την γενικότερη βεβαιότητα για την ύπαρξη της ίδιας της ευπάθειας. Τα απαραίτητα στοιχεία για τον υπολογισμό του σκορ είναι:
  - **Exploit Code Maturity:** Περιγράφει το κατά πόσο είναι πιθανό να γίνει κάποια επίθεση εκμεταλλευόμενη αυτήν την ευπάθεια και βασίζεται στην ύπαρξη τεχνικών και κώδικα για την

εκμετάλλευσή της.

- **Remediation Level:** Τι μηχανισμοί προστασίας κι επιδιόρθωσης της ευπάθειας υπάρχουν (άτυπες προσωρινές μέθοδοι, επίσημες επιδιορθώσεις).
- **Report Confidence:** Κατά πόσο είναι βέβαιο ότι η ευπάθεια υπάρχει γενικότερα και αν υπάρχουν τεχνικές εκμετάλλευσής της.

- **Environmental Score:** Αυτό το σκορ ουσιαστικά είναι ίδιο με το Base Score, αλλά προσαρμοσμένο στα χαρακτηριστικά του εκάστοτε συστήματος, ώστε η ευπάθεια να αξιολογείται πιο αντικειμενικά και συγκεκριμένα.

Στα πλαίσια της εργασίας, χρησιμοποιήθηκε αποκλειστικά το Base Score, αφού αφενός οι ευπάθειες που μελετώνται είναι επιβεβαιωμένες και υπάρχουν τόσο τεχνικές εκμετάλλευσής τους, όσο και τεχνικές αποτροπής τους, επομένως το Temporal Score θα ήταν ίδιο σε όλες κι αφετέρου εξετάζονται πολλά διαφορετικά συστήματα, άρα δεν είναι εφικτό να παραχθεί ένα συγκεκριμένο Environmental Score για κάθε ξεχωριστό σύστημα.

Πίνακας 2.1 Απόδοση σκορ στους εξεταζόμενους μηχανισμούς άμυνας κι ευπάθειες

Μηχνανισμός Άμυνας/Ευπάθεια	**Σχετική επίθεση**	**Σκορ**
HttpOnly cookies	Cross Site Scripting	6,1
Secure cookies	Man in The Middle	6,9
Strict-Transport-Security	Man in The Middle	6,9
X-Frame-Options	Clickjacking	5,4
X-XSS-Protection	Cross Site Scripting	6,1
X-Content-Type-Options	Cross Site Scripting	6,1
Iframe sandboxing	Clickjacking	5,4
CSRF tokens	Cross-Site Request Forgery	4,3
Stale JavaScript Inclusions	JavaScipt Code Injection	-7,2
SSL-Stripping	Man in The Middle	-8,3

Κεφάλαιο 3

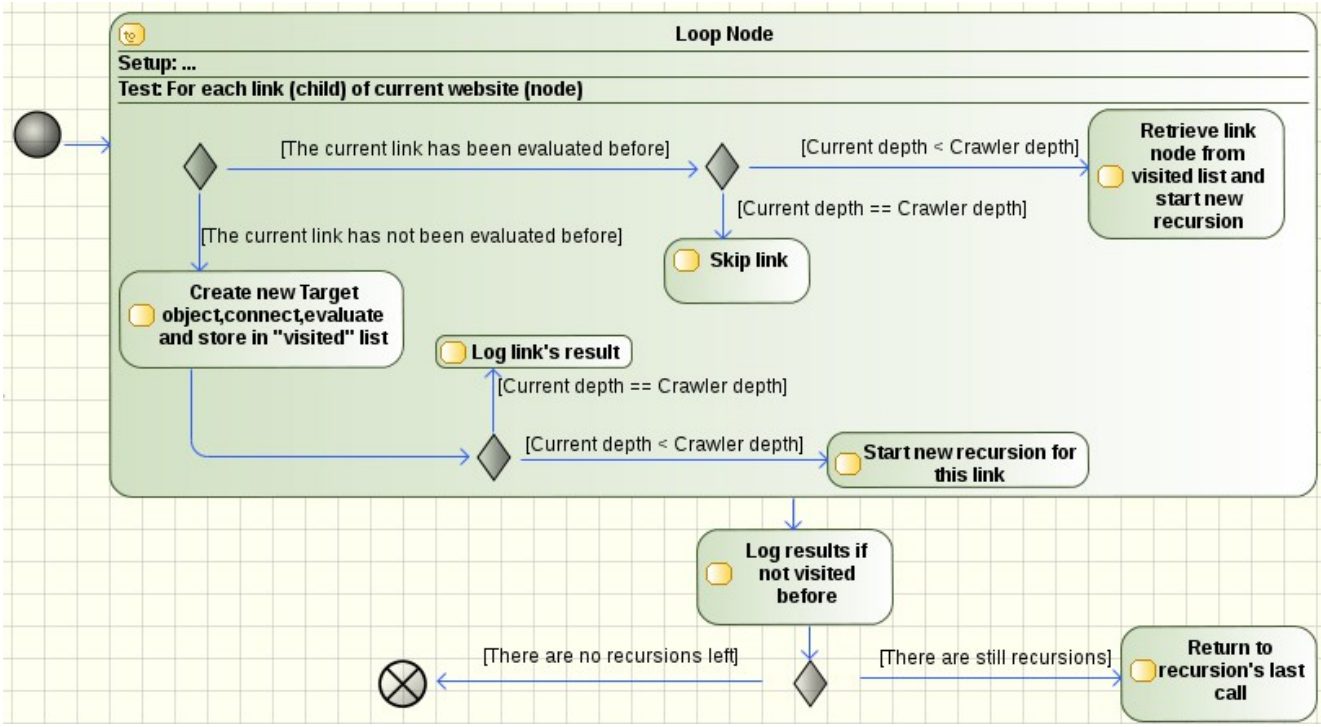
Σχεδιασμός

3.1 Αρχιτεκτονική Εφαρμογής

Η αρχιτεκτονική της παρούσας εργασίας βασίζεται σε 4 βασικά συστατικά στοιχεία που έχουν να κάνουν με την παραμετροποίηση της εκτέλεσης από τον χρήστη, την σύνδεση στις ιστοσελίδες και την συλλογή των απαραίτητων πληροφοριών, την αξιολόγηση των ιστοσελίδων και τέλος με την καταγραφή των αποτελεσμάτων της εκτέλεσης.

Αρχικά, μέσα από την γραμμή εντολών ο χρήστης δίνει τις κατάλληλες παραμέτρους για την εκτέλεση της εφαρμογής, όπως την αρχική διεύθυνση ή το αρχείο με τις αρχικές διευθύνσεις που θα αξιολογηθούν, τις προαιρετικές παραμέτρους για το crawling mode και το βάθος του crawling και την επίσης προαιρετική παράμετρο για την παρουσίαση στατιστικών αποτελεσμάτων της εκτέλεσης. Στην συνέχεια, η εφαρμογή με βάση τις παραπάνω παραμέτρους εκτελείται αναλόγως, συνδέεται σειριακά στην κάθε αρχική ιστοσελίδα και αν εκτελείται σε crawling mode συλλέγει τους εσωτερικούς συνδέσμους της καθεμίας και τους επισκέπτεται και τους αξιολογεί αναδρομικά μέχρι το δοθέν βάθος. Για κάθε σύνδεση η εφαρμογή συλλέγει κι αξιολογεί συγκεκριμένες πληροφορίες και παράγει ένα σκορ που αντιστοιχεί στην συγκεκριμένη ιστοσελίδα, αλλά κι ένα σκορ εμπιστοσύνης που προκύπτει από τα σκορ των εσωτερικών συνδέσμων της.

Στο σχήμα 3.1 φαίνεται η αρχιτεκτονική της εφαρμογής στο crawling mode, ανεξαρτήτως βάθους.



Σχήμα 3.1: Αρχιτεκτονική της εφαρμογής σε crawling mode

3.2 Εργαλεία υλοποίησης εφαρμογής

3.2.1 Βιβλιοθήκες Python

3.2.1.1 urllib2

Η urllib2 είναι μία βιβλιοθήκη της γλώσσας Python, που προσφέρει την δυνατότητα σύνδεσης σε σελίδες μέσω των πρωτοκόλλων HTTP, HTTPS και FTP. Η βιβλιοθήκη μέσω ποικίλων συναρτήσεων και κλάσεων κάνει εύκολη την διαχείριση των συνδέσεων σε ιστοσελίδες και την παραμετροποίηση των αιτήσεων με τα κατάλληλα HTTP headers.

Πιο συγκεκριμένα, η παραμετροποίηση των αιτήσεων δίνει την δυνατότητα να ορίσουμε HTTP headers όπως το “User-agent”, έτσι ώστε να μην είναι φανερή η πραγματική ταυτότητα του crawler, αλλά να φαίνεται ως ένας κανονικός browser. Ακόμη επιτρέπει τον ορισμό ενός timeout, δηλαδή μιας χρονικής τιμής που αν ξεπεραστεί, η διαδικασία σύνδεσης διακόπτεται, κάτι αρκετά χρήσιμο για την λειτουργία ενός web crawler που χρήζει γρήγορης και συνεχούς εκτέλεσης. Επιπλέον, μέσω της βιβλιοθήκης urllib2 είναι εφικτή

η ανάκτηση των HTTP headers που επιστρέφονται από την ιστοσελίδα, καθώς επίσης και η ανάκτηση του HTML κώδικα της επιστρεφόμενης σελίδας, δύο βασικά συστατικά στοιχεία που αξιολογούνται από την παρούσα εργασία.

3.2.1.2 lxml.html

Η βιβλιοθήκη αυτή εστιάζει στην διαχείριση δεδομένων τύπου XML και πιο συγκεκριμένα σε δεδομένα μορφοποιημένα ως HTML. Παίρνοντας ως παράμετρο μία σειρά χαρακτήρων, όπως τον HTML κώδικα μιας ιστοσελίδας, μπορεί να τον μετατρέψει στην κατάλληλη δομημένη μορφή έτσι ώστε στην συνέχεια να διασχιστεί και να συλλεχθούν τα δεδομένα που χρειάζονται με την χρήση π.χ. XPath, κάτι εξαιρετικά χρήσιμο σε έναν crawler ή scraper που συλλέγει συγκεκριμένες πληροφορίες κι εσωτερικούς συνδέσμους ιστοσελίδων για να συνεχίσει την εκτέλεσή του.

3.2.1.3 JSON

Το JSON είναι ένα ελαφρύ πρότυπο ανταλλαγής δεδομένων, το οποίο είναι εύκολο για τους ανθρώπους να το διαβάσουν και να το ερμηνεύσουν κι ακόμη πιο εύκολο για τους υπολογιστές να το αναλύσουν και να το παράξουν. Ακόμη, αξίζει να σημειωθεί ότι πρόκειται για ένα ανεξάρτητο από γλώσσες προγραμματισμού πρότυπο, αν και βασίζεται σε ένα υποσύνολο της γλώσσας προγραμματισμού JavaScript, κάτι που μαζί με τα άλλα χαρακτηριστικά του το καθιστούν κατάλληλο για καταγραφή κι ανάλυση δεδομένων. [15]

Λόγω αυτών του των ιδιοτήτων το JSON αρμόζει για την καταγραφή μεγάλου όγκου δεδομένων σε εξωτερικά αρχεία, ώστε να παραμένουν ανοικτά προς επανάληψη ή και επαναξιολόγηση από οποιονδήποτε, ανεξάρτητα από γλώσσα προγραμματισμού, κάτι που η παρούσα εργασία επιδιώκει.

Η ομώνυμη βιβλιοθήκη της Python συμβάλλει στην δημιουργία και τροποποίηση δεδομένων τύπου JSON. Συγκεκριμένα, μπορεί να μορφοποιήσει δεδομένα και τύπους δεδομένων της Python, όπως ένα λεξικό (dictionary), σε JSON μορφή, αλλά και αντίστροφα να διαβάσει αντικείμενα JSON και να τα ερμηνεύσει σε δικούς της τύπους δεδομένων ή αντικείμενα.

Κεφάλαιο 4

Υλοποίηση

Σε αυτό το κεφάλαιο περιγράφεται η υλοποίηση της εφαρμογής που αναπτύχθηκε στο πλαίσια της παρούσας πτυχιακής εργασίας. Στόχος της εφαρμογής είναι, όπως αναφέρθηκε παραπάνω, η συλλογή κι αξιολόγηση δημόσια διαθέσιμων πληροφοριών συσχετισμένων με την ασφάλεια μίας ή περισσότερων ιστοσελίδων και η εξαγωγή μεμονωμένων αλλά και στατιστικών αποτελεσμάτων της εκτέλεσης. Στο κεφάλαιο θα περιγραφούν αναλυτικά οι λεπτομέρειες υλοποίησης της παραμετροποίησης της εφαρμογής, της σύνδεσης στις ιστοσελίδες, καθώς και ο αλγόριθμος που υλοποιήθηκε για το crawling, η συλλογή κι αξιολόγηση των διάφορων πληροφοριών (μηχανισμοί ασφάλειας, ευπάθειες), η καταγραφή και οι μορφοποιήσεις των αποτελεσμάτων και τέλος η εξαγωγή στατιστικών αποτελεσμάτων από τα επιμέρους αποτελέσματα.

4.1 Παραμετροποίηση κι εκκίνηση εκτέλεσης – Η κλάση *Init*

Η εφαρμογή υλοποιήθηκε έτσι ώστε να δέχεται ως είσοδο συγκεκριμένες παραμέτρους από την γραμμή εντολών, άλλες υποχρεωτικές κι άλλες προαιρετικές. Συγκεκριμένα, οι παράμετροι που δέχεται η εφαρμογή είναι οι εξής:

- **-u, --url** Υποχρεωτική, εκτός αν έχει οριστεί το **-f**. Η διεύθυνση της ιστοσελίδας που θα αξιολογηθεί και το σημείο εκκίνησης του crawling (παράμετρος **-c**).
- **-f, --file** Υποχρεωτική, εκτός αν έχει οριστεί το **-u**. Το αρχείο που περιέχει το σύνολο των ιστοσελίδων που θα αξιολογηθούν και θα αποτελέσουν τα σημεία εκκίνησης του crawling

(παράμετρος **-c**).

- **[-c, --crawl]** Προαιρετική. Ενεργοποίηση του crawling mode. Οι αρχικές διευθύνσεις που ορίστηκαν από τις παραμέτρους **-u** ή **-f** θα αποτελέσουν τα σημεία εκκίνησης του crawling.
- **[-d, --depth]** Προαιρετική. Απαιτεί το **-c**. Ορισμός του βάθους του crawling.
- **[-v, --verbose]** Προαιρετική. Ορίζει αν η εφαρμογή θα τυπώνει κατά την διάρκεια της εκτέλεσης τα ευρήματά της στην έξοδο.
- **[-s, --stats]** Προαιρετική. Ορίζει αν μετά την εκτέλεση θα υπολογιστούν στατιστικά αποτελέσματα με βάση τα ευρήματά της.

Η κλάση ***Init*** είναι υπεύθυνη για κάποιες βασικές ενέργειες πριν την εκκίνηση της αξιολόγησης, όπως την δημιουργία του φακέλου */logs*, όπου αποθηκεύονται όλα τα αρχεία με τα αποτελέσματα της εφαρμογής, την δημιουργία των αρχείων καταγραφής των αποτελεσμάτων αλλά και την ερμηνεία των παραμέτρων που έδωσε ο χρήστης μέσα από την γραμμή εντολών και την εκτέλεση των αντίστοιχων ενεργειών. Ουσιαστικά χωρίζει την παραμετροποίηση της εφαρμογής σε δύο μεγάλες κατηγορίες: αν έχει οριστεί η παράμετρος **-u**, όπου έχει οριστεί μία μόνο αρχική διεύθυνση και αν έχει οριστεί η παράμετρος **-f**, όπου ορίζονται περισσότερες. Αξίζει να σημειωθεί ότι ακόμη κι αν ο χρήστης έχει ορίσει μία σελίδα με το πρωτόκολλο HTTP, π.χ. *http://www.example.com*, τότε το πρόθεμα *http://* θα μετατραπεί σε *https://*, έτσι ώστε αφενός να εντοπίζονται πεδία που χωρίς αυτό το πρωτόκολλο δεν θα ήταν ορατά, π.χ. secure cookies, και αφετέρου για να επιβεβαιώνεται ότι η ιστοσελίδα υποστηρίζει ασφαλείς συνδέσεις. Σε κάθε περίπτωση, για κάθε αρχική διεύθυνση καλεί την μέθοδο ***init_evaluation(url)***, η οποία με την σειρά της ανάλογα με το αν η εφαρμογή τρέχει σε crawling mode (**-c**) ή όχι δημιουργεί ένα στιγμιότυπο της αντίστοιχης κλάσης, όπως φαίνεται στο παρακάτω απόσπασμα κώδικα.

```
#Check the specified mode and start the evaluation process for the target URL
def init_evaluation(self, url):
    if self.args.crawl:
        if self.args.depth:
            crawl = Crawler(url, self.args.depth, self.args.verbose)
        else:
            crawl = Crawler(url, 1, self.args.verbose) #Default crawl depth is 1
        if not crawl.start():
            return False
    else:
        rootnode = Target(url, None, self.args.verbose)
        rootnode.analyse()
        #In case the connection to the target failed, do not write to the json file
        if rootnode.scored == None:
            return False
        write_json_result(rootnode.url, rootnode.final, None)
    return True
```

4.2 Σύνδεση κι αξιολόγηση – Η κλάση *Target*

Την σύνδεση σε κάθε ιστοσελίδα αναπαριστά ένα στιγμιότυπο της κλάσης ***Target***. Συγκεκριμένα η κλάση αυτή είναι υπεύθυνη για τα χαρακτηριστικά σύνδεσης στην ιστοσελίδα, όπως τα πεδία HTTP Header που θα αποσταλούν σε αυτήν, την ίδια την σύνδεση και προσαρμοσμένες επαναλήψεις της σε περίπτωση αποτυχίας ή σφάλματος, την ανάκτηση των διάφορων δεδομένων από την ιστοσελίδα, δηλαδή τα HTTP Headers που επιστρέφονται και τα σχετικά στοιχεία από τον HTML κώδικά της και τέλος την μορφοποίηση και αξιολόγηση τους. Ακόμη, η κλάση αυτή είναι υπεύθυνη για την συλλογή των συνδέσμων της σελίδας που οδηγούν σε εξωτερικές διευθύνσεις και που χρησιμοποιούνται κατά το crawling mode για την δημιουργία νέων αντικειμένων Target. Βασικό πεδίο της κλάσης είναι το λεξικό ***final***, το οποίο γεμίζει κατά την ανάκτηση των δεδομένων και περιέχει όλες τις τιμές των αντίστοιχων πεδίων που συλλέχθηκαν. Η μορφή του είναι η ακόλουθη:

```
#Dictionary containing all retrieved header fields, the HTML related results, the website's final score, trust score, parent node
self.final = {"server":"", "set-cookie":[], "strict-transport-security":"",
    "x-frame-options":"", "x-xss-protection":"", "x-content-security-policy":"",
    "content-security-policy":"", "x-content-type-options":"",
    "x-powered-by":"", "httponly":0, "secure-cookies":0, "stalejs":None, "iframes":None,
    "csrf":None, "ssl-strip":None, "score":0,
    "trust":None, "parent":""};
```

Η κύρια μέθοδος της κλάσης είναι η ***analyse()***, η οποία ουσιαστικά καλεί όλες τις υπόλοιπες μεθόδους αξιολόγησης, αφού καλέσει την μέθοδο σύνδεσης.

4.2.1 Σύνδεση

Η μέθοδος που πραγματοποιεί την σύνδεση με την ιστοσελίδα είναι η ***connect(data = None)***. Σε πρώτη φάση, προετοιμάζει την αίτηση σύνδεσης ορίζοντας τα πεδία HTTP Header που θα αποσταλούν, όπως τα *User-agent*, *Accept*, *Accept-Language*, *Connection* και *Referer*. Ακόμη, αν η παράμετρος *data* έχει οριστεί, τότε η αίτηση θα γίνει μέσω της HTTP μεθόδου POST, ειδάλλως – και το πιο σύνηθες – μέσω GET. Εν συνεχεία, πραγματοποιείται η προσπάθεια σύνδεσης στην σελίδα, η οποία έχει ένα χρονικό περιθώριο 10 δευτερολέπτων (timeout) και ανακτώνται τα HTTP Header πεδία που επιστρέφονται και ο HTML κώδικας

της σελίδας. Σε περίπτωση σφάλματος, η μέθοδος ξαναδοκιμάζει την σύνδεση με δύο διαφορετικούς τρόπους. Πρώτα, προσπαθεί να συνδεθεί μέσω του πρωτοκόλλου HTTP, αντί για HTTPS. Αν η επανασύνδεση επιτύχει υποδεικνύεται ότι η ιστοσελίδα δεν υποστηρίζει το πρωτόκολλο HTTPS κι επομένως ούτε ασφαλείς συνδέσεις. Σε περίπτωση αποτυχίας και πάλι, δοκιμάζει με την αφαίρεση ή πρόσθεση του προθέματος "*www.*", διότι αρκετές ιστοσελίδες δεν κάνουν ανακατεύθυνση από το "www.example.com" στο "*example.com*", για παράδειγμα. Εν τέλει, ελέγχει αν η ιστοσελίδα ανακατεύθυνε σε κάποια άλλη διεύθυνση και ανανεώνει την διεύθυνση του αντικειμένου τύπου ***Target*** ανάλογα. Ο βασικός κορμός της μεθόδου είναι ο εξής.

```
125  req_headers = {"User-agent":Target.user_agent, "Accept":Target.accept, "Accept-Language":Target.lang,"Connection":Target.conn,"Referer"
126
127  try:
128      #Make the request to the given url
129      req = urllib2.Request(self.url, data, req_headers)
130      response = urllib2.urlopen(req, timeout=10)
131      self.htmlre = response.read()  #html response
132      self.headers = response.info()  #Get the headers
133  except urllib2.HTTPError as err:
134      self.finish_progress(finished, progress_thread)
135      self.printmsg("[!] The request at: "+self.url+" did not succeed\n    Error code: "+str(err.code))
136      #Retry connection with http instead of https and also by including/removing the "www" prefix
137      self.retry_conn()
138      return self.headers
```

4.2.2 Ανάλυση δεδομένων κι αξιολόγηση

Για την αξιολόγηση των δεδομένων που ανακτήθηκαν από την ιστοσελίδα, καλείται σειριακά ένας αριθμός μεθόδων μέσα από την μέθοδο ***analyse()***, όπου η καθεμία είναι υπεύθυνη για τον έλεγχο της τιμής ή της ύπαρξης κάποιου στοιχείου, π.χ. για την ύπαρξη cookies και για το αν αυτά έχουν οριστεί με την ιδιότητα *httponly*. Τα στοιχεία αυτά έχουν αποθηκευτεί στο λεξικό ***final***, ώστε να είναι συγκεντρωμένα και να μπορούν αργότερα να καταγραφούν ομαδικά κι εύκολα σε μορφοποίηση JSON. Οι μέθοδοι που καλούνται χωρίζονται σε δύο υποκατηγορίες, αυτές που είναι συσχετισμένες με τα HTTP Headers κι αυτές που αφορούν στοιχεία από τον HTML κώδικα της σελίδας. Αξίζει να σημειωθεί ότι οι μέθοδοι της δεύτερης κατηγορίας καλούνται μετά την επιτυχή μετατροπή του HTML κώδικα της σελίδας σε δομημένη XML μορφή με την χρήση της βιβλιοθήκης ***lxml.html***, ώστε τα επιθυμητά στοιχεία να μπορούν να ανακτηθούν με την χρήση XPath. Πιο συγκεκριμένα:

- Μέθοδοι για τον έλεγχο των HTTP Headers:
    - ***eval_cookies()***: Ελέγχει την ύπαρξη cookies μέσω του πεδίου HTTP Header *Set-Cookie* κι αν σε τουλάχιστον ένα από αυτά έχει δηλωθεί η ιδιότητα *httponly* και *secure* αντίστοιχα.

- ***versions():*** Ελέγχει την ύπαρξη των πεδίων *X-Powered-By* και *Server*, τα οποία δίνουν πληροφορίες για την έκδοση της γλώσσας ή του framework που εκτελείται στον server και για την έκδοση του server αντίστοιχα. Η συγκεκριμένη μέθοδος δεν αποδίδει κάποιο σκορ διότι τα πεδία που αφορά δεν σχετίζονται με κάποια ευπάθεια, απλά ανακτά αυτές τις πληροφορίες προς γνώση του χρήστη της εφαρμογής.
- ***hsts():*** Ελέγχει την ύπαρξη του πεδίου *Strict-Transport-Security*.
- ***x_frame_opts():*** Αφορά το πεδίο *X-Frame-Options*, αν υπάρχει κι αν είναι ενεργό.
- ***xXss():*** Αφορά το πεδίο *X-XSS-Protection*.
- ***x_content_type():*** Αφορά το πεδίο *X-Content-Type-Options*.
- ***x_policy():*** Τα πεδία που ελέγχονται είναι τα *Content-Security-Policy* και *X-Content-Security-Policy*.

• Μέθοδοι που αφορούν HTML στοιχεία:

- ***check_js_includes():*** Αφού ανακτήσει τους παρόχους των εξωτερικών αρχείων JavaScript που εμπεριέχονται στην σελίδα, ελέγχει αν η διεύθυνσή τους είναι υπαρκτή κι αντιστοιχεί σε κάποια διεύθυνση IP. Εάν υπάρχει έστω κι ένας πάροχος που δεν πέρασε τον παραπάνω έλεγχο, τότε αποδίδεται ένα αρνητικό σκορ. Αυτό διότι πρόκειται για μία άμεση ευπάθεια που μπορεί να την εκμεταλλευτεί κάποιος επιτιθέμενος.
- ***check_iframes():*** Η μέθοδος αυτή συλλέγει τα <*iframe*> στοιχεία από τον HTML κώδικα κι ελέγχει αν έχει η οριστεί η ιδιότητά *sandbox*.
- ***check_csrf_token():*** Εδώ ανακτώνται συγκεκριμένα στοιχεία φορμών από την ιστοσελίδα. Ειδικότερα, η μέθοδος προσπαθεί να συλλέξει όσα στοιχεία είναι τύπου <*input*>, είναι κρυμμένα, είναι δηλαδή της μορφής <*input type=”hidden”*>, το όνομά τους περιέχει κάποια σχετική λέξη κλειδί, όπως π.χ. “*csrf*”, “*token”, “key”* κ.α. και η τιμή τους είναι μεγαλύτερη από 20 χαρακτήρες. Αν βρεθεί κάποιο στοιχείο που να πληροί όλα τα παραπάνω γίνεται η παραδοχή ότι πρόκειται για ένα CSRF token.
- ***check_ssl_stripping():*** Ελέγχεται η ιδιότητα *action* των φορμών που περιέχονται στην σελίδα και πιο συγκεκριμένα αν δείχνει σε κάποια άλλη σελίδα με το πρωτόκολλο HTTPS. Αν η παραπάνω συνθήκη ισχύει και ισχύει ότι η τρέχουσα σελίδα χρησιμοποιεί HTTP, τότε αποδίδεται ένα αρνητικό σκορ διότι και πάλι πρόκειται για μία ευπάθεια.

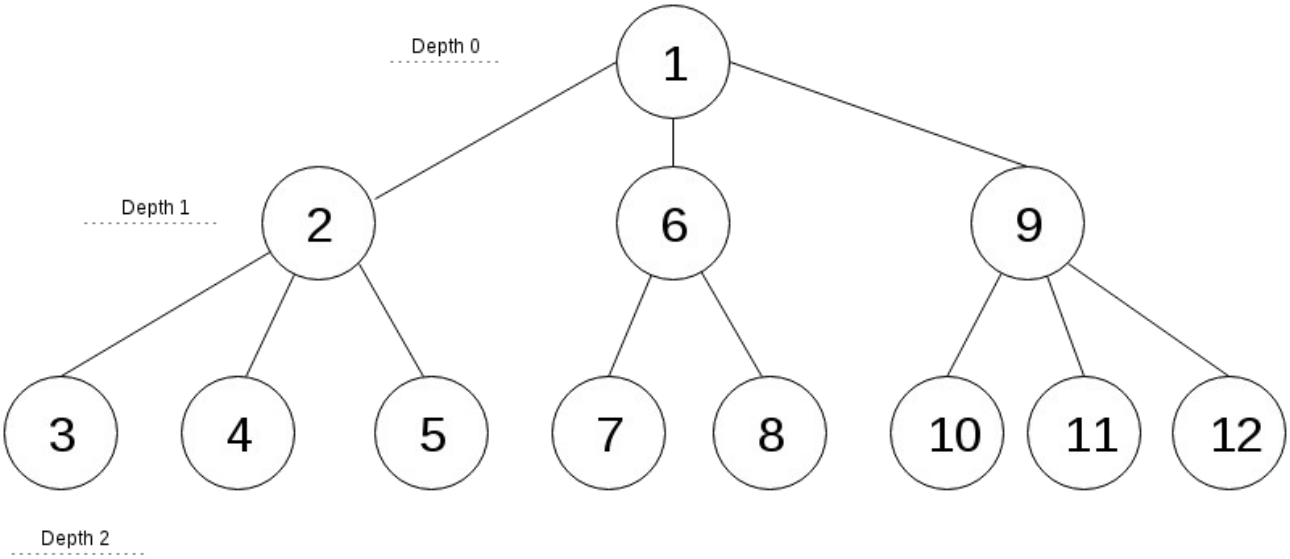
4.3 Crawling mode – Η κλάση *Crawler*

Όταν ο χρήστης δίνει από την γραμμή εντολών την παράμετρο ***-c***, η εφαρμογή εκτός από την αρχική διεύθυνση, ή τις αρχικές διευθύνσεις, αξιολογεί και τους συνδέσμους της που οδηγούν σε άλλους ιστοτόπους. Μάλιστα, αν το βάθος του crawling οριστεί πάνω από 1, που είναι η προεπιλεγμένη τιμή του, για παράδειγμα 2, τότε ανακτώνται κι αξιολογούνται και οι σύνδεσμοι του κάθε συνδέσμου της αρχικής σελίδας, αν είναι 3 τότε οι σύνδεσμοι του καθενός από τους τελευταίους συνδέσμους κ.ο.κ.

Η κλάση που είναι υπεύθυνη για την σειρά εκτέλεσης και την ίδια την εκτέλεση του crawling είναι η ***Crawler***. Το κάθε αντικείμενο της κλάσης αυτής αρχικοποιείται με την αρχική διεύθυνση και το δοθέν βάθος. Οι βασικές της μέθοδοι είναι η ***start()*** και η ***visit_children(node, depth)***, στην οποία το όρισμα ***node*** είναι ένα αντικείμενο της κλάσης ***Target*** που περιγράφηκε παραπάνω και το ***depth*** το τρέχον βάθος που έχει φτάσει το crawling.

Ειδικότερα, στην μέθοδο ***start()***, δημιουργείται ένα αντικείμενο της κλάσης ***Target*** για την αρχική διεύθυνση και καλείται η μέθοδός του ***analyse()***. Εν συνεχεία, καλείται η μέθοδος ***visit_children(hdroot, 1)***, με κόμβο το αντικείμενο που δημιουργήθηκε για την αρχική διεύθυνση και βάθος 1. Αφού κληθεί η μέθοδος, ξεκινάει μία επανάληψη για όλους τους συνδέσμους του δοθέν κόμβου. Σε κάθε επανάληψη δημιουργείται ένα νέο αντικείμενο ***tmpnode*** της κλάσης ***Target*** για κάθε σύνδεσμο και αν η αξιολόγησή του είναι επιτυχής, η διεύθυνσή του και το ίδιο το αντικείμενο διατηρούνται σε μία καθολική στατική μεταβλητή της μορφής ***Crawler.visited['exampleurl.com'] = [Target(exampleurl)]***, έτσι ώστε να αποφεύγεται η επίσκεψη του ίδιου ιστοτόπου σε περίπτωση που ξαναβρεθεί ως σύνδεσμος σε κάποια επανάληψη. Πριν το τέλος της κάθε επανάληψης ελέγχεται αν η εκτέλεση έχει φτάσει στο δοθέν βάθος. Αν όχι, τότε η μέθοδος ***visit_children(tmpnode, depth+1)*** καλείται αναδρομικά με βασικό κόμβο αυτήν την φορά τον τελευταίο σύνδεσμο που αξιολογήθηκε. Σε περίπτωση που το βάθος έχει φτάσει στο όριό του, ο σύνδεσμος που αξιολογήθηκε μορφοποιείται κατάλληλα σε JSON και καταγράφεται στο αρχείο αποτελεσμάτων. Όταν η επανάληψη για έναν κόμβο τελειώσει υπολογίζεται το ***trust*** του, δηλαδή ο μέσος όρος των σκορ των συνδέσμων του που αντιπροσωπεύει ένα σκορ εμπιστοσύνης του ιστοτόπου προς τους άλλους κι εν τέλει ο κόμβος αφού μετατραπεί κατάλληλα σε JSON μορφή καταγράφεται στο αρχείο αποτελεσμάτων.

Με τα βήματα που περιγράφηκαν παραπάνω ουσιαστικά ακολουθείται μία DFS (Depth First Search) [16] προσέγγιση και κατασκευάζεται ένα δέντρο που δημιουργείται με την σειρά που φαίνεται παρακάτω.



Σχήμα 4.1 Σχηματισμός του δέντρου του crawling

4.4 Καταγραφή και παρουσίαση αποτελεσμάτων

Για την καταγραφή των αποτελεσμάτων χρησιμοποιούνται δύο τρόποι. Αρχικά, τα αποτελέσματα καταγράφονται σε ένα αρχείο σε φυσική γλώσσα κι αν έχει δοθεί και η παράμετρος ***-v*** τυπώνονται με τον ίδιο τρόπο και στην κύρια έξοδο. Ένα παράδειγμα εξόδου της εκτέλεσης είναι το εξής:

```
root@localhost:~/Desktop/ptyxiakh/ptx# ./headersec.py -u twitter.com -v
[!] Initiating....

[!] Connecting @ https://twitter.com

Examining headers...
[+] 1 HttpOnly cookie(s) found out of 2
[+] 1 Secure cookie(s) found out of 2
[-] Server name and/or version revealed: tsa_b
[+] HTTP Strict-Transport-Security is set
[+] X-Frame-Options is set to: SAMEORIGIN
[+] X-XSS-Protection is enabled
[+] X-Content-Type-Options is set to: nosniff
[+] (X-)Content-Security-Policy is set

Examining HTML response...
[+] No stale JavaScript includes found
[+] CSRF token found

[~] https://twitter.com scored: 49.0 points
```

Σχήμα 4.2 Παράδειγμα εξόδου απλής εκτέλεσης

Ο δεύτερος τρόπος είναι η καταγραφή των αποτελεσμάτων σε μορφή JSON και πάλι σε ένα τοπικό αρχείο. Το αρχείο αυτό στην συνέχεια χρησιμοποιείται για την εξαγωγή των στατιστικών αποτελεσμάτων της εκτέλεσης. Ένα δείγμα ενός τέτοιου αρχείου είναι το ακόλουθο:

```
    "https://twitter.com": {
        "parent": "https://lifo.gr",
        "x-xss-protection": "1; mode=block",
        "iframes": 0,
        "x-content-type-options": "nosniff",
        "content-security-policy": "default-src 'self'; connect-src 'self' https://upload.twitter.com https:/
        "x-powered-by": "",
        "set-cookie": [
            "_twitter_sess=BAh7CSIKZmxhc2hJQzonQWN0aW9uQ29udHJvbGxlcjo6Rmxhc2g6OkZsYXNo%250ASGFzaHsABjoKQHVz
            "guest_id=v1%3A144035320100486729; Domain=.twitter.com; Path=/; Expires=Tue, 22-Aug-2017 18:06:41
        ],
        "strict-transport-security": "max-age=631138519",
        "server": "tsa_b",
        "ssl-strip": null,
        "stalejs": 0,
        "score": 49,
        "csrf": 1,
        "secure-cookies": 1,
        "trust": null,
        "x-frame-options": "SAMEORIGIN",
        "x-content-security-policy": "",
        "httponly": 1
    },
```

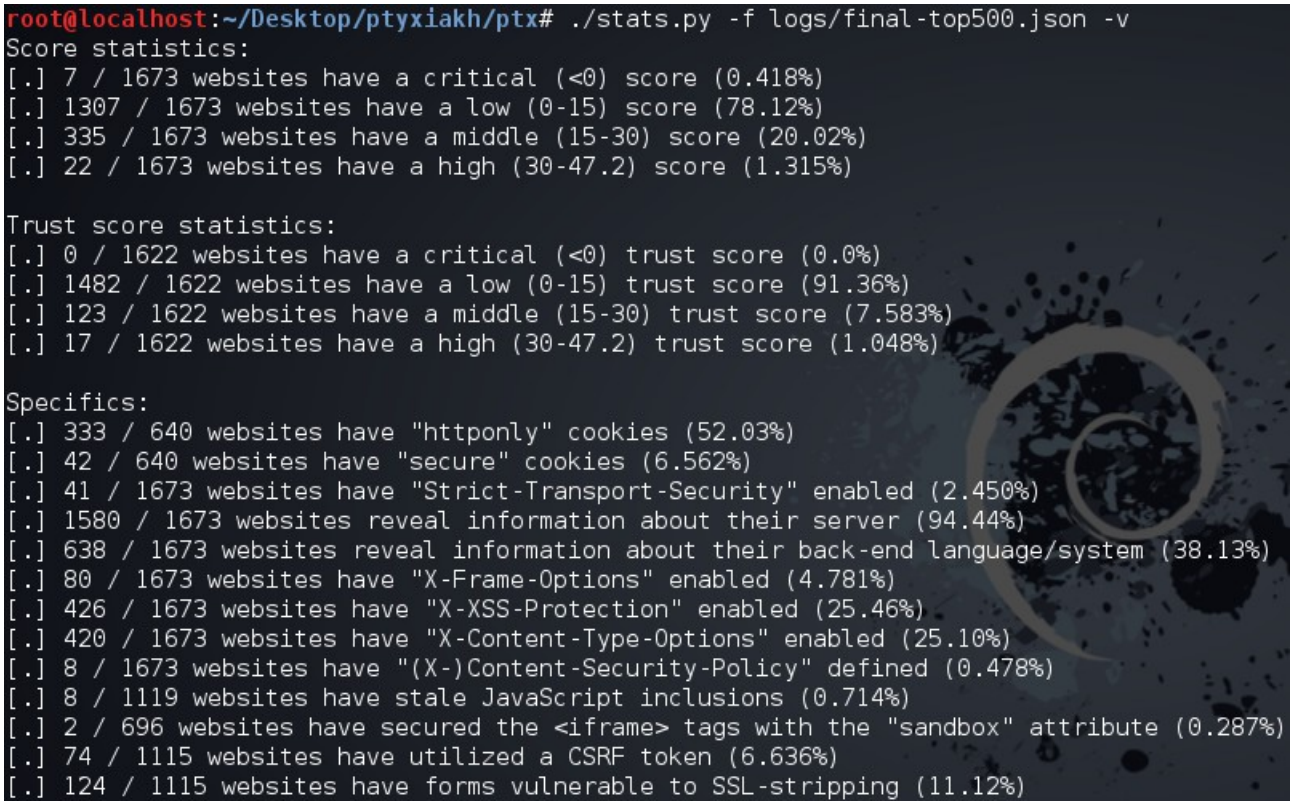

Σχήμα 4.3 Παράδειγμα αντικειμένου JSON

4.4.1 Στατιστικά αποτελέσματα – Το αρχείο *stats.py*

Για την εξαγωγή στατιστικών αποτελεσμάτων της εκτέλεσης, που έχει νόημα κυρίως σε μία εκτέλεση μεγάλης κλίμακας, καλούνται μέθοδοι από ένα εξωτερικό αρχείο – το ***stats.py*** – που μπορεί να εκτελεστεί και μόνο του, ανεξάρτητα από την εκτέλεση της βασικής εφαρμογής. Είσοδος είναι ένα αρχείο JSON όπως το παραπάνω, που περιέχει τα αποτελέσματα μίας εκτέλεσης της κύριας εφαρμογής.

Για την παραγωγή των στατιστικών αποτελεσμάτων υπολογίζονται αρχικά διάφορα σύνολα μέσα από το αρχείο JSON της εισόδου. Πέρα από το βασικό σύνολο όλων των ιστοσελίδων που αξιολογήθηκαν, υπολογίζεται το σύνολο των ιστοσελίδων που χρησιμοποιούν cookies, το σύνολο που έχουν συμπεριλάβει JavaScript κώδικα από τρίτους, εξωτερικούς παρόχους, το σύνολο των ιστοσελίδων που κάνουν χρήση iframes και τέλος το σύνολο των ιστοσελίδων που περιέχουν κάποια HTML φόρμα. Τα σύνολα αυτά χρειάζονται έτσι ώστε τα τελικά ποσοστά που υπολογίζονται να αφορούν τις αντίστοιχες ιστοσελίδες και να είναι αντικειμενικά. Δεν θα είχε νόημα να συμπεριληφθεί, για παράδειγμα, στο ποσοστό των ιστοσελίδων που έχουν την ιδιότητα *sandbox* στα στοιχεία iframe μία σελίδα που δεν περιέχει καθόλου iframes.

Τα τελικά ποσοστά που υπολογίζονται εμφανίζονται στην κύρια έξοδο, αλλά και αποθηκεύονται με την ίδια μορφή σε ένα αρχείο. Ακόμη, δίνεται η δυνατότητα εξαγωγής αποτελεσμάτων μόνο για τις αρχικές ιστοσελίδες. Ένα παράδειγμα εκτέλεσης είναι το ακόλουθο:



Σχήμα 4.4 Παράδειγμα εξόδου στατιστικών αποτελεσμάτων

Κεφάλαιο 5

Αποτελέσματα

5.1 Use Case: Αξιολόγηση των 500 κορυφαίων σε επισκεψιμότητα ιστοσελίδων στην Ελλάδα

Για την ανάδειξη της λειτουργικότητας της εφαρμογής που αναπτύχθηκε στην παρούσα εργασία, έπρεπε αφενός να επιλεχθεί ένα μεγάλο εύρος ιστοσελίδων ώστε να φανεί η λειτουργικότητα του crawler κι αφετέρου να επιλεχθεί ένα σύνολο το οποίο να έχει μία ουσιαστική σημασία, έτσι ώστε να εξαχθούν κάποια χρήσιμα συμπεράσματα. Για τον λόγο αυτό, επιλέχθηκαν σαν σύνολο οι 500 κορυφαίες σε επισκεψιμότητα ιστοσελίδες στην Ελλάδα. Αξίζει να σημειωθεί ότι δεν πρόκειται μόνο για ελληνικές ιστοσελίδες, αλλά και ιστοσελίδες από το εξωτερικό.

5.1.1 Χαρακτηριστικά εκτέλεσης

Οι διευθύνσεις των ιστοσελίδων ανακτήθηκαν αυτοματοποιημένα από το Alexa.com [17], που παρέχει ακριβώς αυτήν την λίστα διευθύνσεων, καθώς κι άλλες που θα ήταν χρήσιμες στην εφαρμογή, και αποθηκεύτηκαν στο αρχείο *500gr.txt*. Τα αποτελέσματα αποθηκεύτηκαν στα αρχεία *500gr.log* και *500gr.json*, σε φυσική γλώσσα και σε μορφοποίηση JSON αντίστοιχα. Η εφαρμογή εκτελέστηκε σε crawling mode, βάθους 1. Αυτό σημαίνει ότι από τις αρχικές διευθύνσεις που συλλέχθηκαν κι αποθηκεύτηκαν στο αρχείο, αξιολογήθηκαν οι ίδιες καθώς και οι σύνδεσμοι που περιείχε η καθεμία. Επιλέχθηκε αυτό το βάθος αφενός γιατί ο αριθμός των σελίδων θα ήταν αρκετός έτσι ώστε να εξαχθούν κάποια συμπεράσματα και να αναδειχθεί η λειτουργικότητα της εφαρμογής κι αφετέρου για να είναι ανεκτός ο χρόνος εκτέλεσης. Εν

ολίγοις, για να εξαχθούν ικανοποιητικά αποτελέσματα στον βέλτιστο δυνατό χρόνο.

Επίσης, προκειμένου να απαλειφθούν πιθανά διπλότυπα αποτελέσματα, τα οποία δεν είναι πάντα δυνατό να εντοπίζονται κατά την εκτέλεση λόγω ανακατευθύνσεων, τα αποτελέσματα JSON πέρασαν από ένα φίλτρο, ώστε να παραμείνουν μοναδικές διευθύνσεις. Εν συνεχεία, προκειμένου να απομονωθούν τα αποτελέσματα που αφορούν μόνο ελληνικές ιστοσελίδες, ώστε να υπάρξει και μία παρουσίαση των δεδομένων που αφορούν το ελληνικό διαδίκτυο αποκλειστικά, φιλτράροντας και πάλι τα αποτελέσματα αποθηκεύτηκαν όλες οι ιστοσελίδες που έχουν την κατάληξη *.gr*.

5.2 Αποτελέσματα εκτέλεσης

Μετά το πέρας της εκτέλεσης η εφαρμογή είχε επισκεφθεί κι αξιολογήσει συνολικά 2661 ιστοτόπους, εκ των οποίων τουλάχιστον οι 1182 ήταν ελληνικοί (είχαν την κατάληξη *.gr*).

Πίνακας 5.1 Σκορ που αποδόθηκαν κατά την εκτέλεση

Κατηγορία Σκορ	**Αριθμός ιστοσελίδων**	**Ποσοστό Ιστοσελίδων (%)**
Critical (<0)	54	2,029
Low (0-15)	2428	91,24
Mid (15-30)	150	5,636
High (30-47,2)	29	1,089

Αν και τα σκορ των ιστοσελίδων είναι προσεγγιστικά και μεγαλύτερη βάση θα έπρεπε να δίνεται στα συγκεκριμένα συλλεγμένα στοιχεία για να εξαχθεί ένα πιο αντικειμενικό συμπέρασμα για το σύνολο των ιστοσελίδων, έχει αξία να παρατηρήσουμε ότι, έστω και προσεγγιστικά, η πλειοψηφία των ιστοσελίδων που αξιολογήθηκαν, περίπου 9 στις 10, έχουν ένα σκορ χαμηλής τάξεως, μεταξύ 0 και 15. Αντίθετα, μόλις ένα πολύ μικρό ποσοστό έχει ένα αρνητικό σκορ, άρα έχει κάποια επιβεβαιωμένη ευπάθεια και όμοια πάλι ένα μικρό ποσοστό έχει χρησιμοποιήσει αρκετούς μηχανισμούς ασφαλείας ώστε να αποκτήσει ένα υψηλό σκορ. Λίγο περισσότερες, αλλά και πάλι σχετικά λίγες, είναι οι ιστοσελίδες εκείνες που έχουν ένα μεσαίο σκορ.

Πίνακας 5.2 Σκορ εμπιστοσύνης που αποδόθηκαν στις αρχικές ιστοσελίδες

Κατηγορία Σκορ	**Αριθμός ιστοσελίδων**	**Ποσοστό Ιστοσελίδων (%)**
Critical (<0)	0	0
Low (0-15)	175	57,18
Mid (15-30)	111	36,27

High (30-47,2)	20	6,535

Σε ό,τι αφορά την εμπιστοσύνη των αρχικών ιστοσελίδων προς τις άλλες ιστοσελίδες που περιέχουν ως συνδέσμους, παρατηρείται ότι η πλειοψηφία έχει ένα χαμηλό σκορ εμπιστοσύνης κι επίσης αρκετές ιστοσελίδες ένα μεσαίας τάξεως σκορ. Αξιοσημείωτο, είναι ότι καμία από τις αρχικές ιστοσελίδες δεν έχει κάποιο αρνητικό σκορ εμπιστοσύνης.

Πίνακας 5.3 Ευρήματα σχετικά με μηχανισμούς άμυνας κι ευπάθειες

Μηχανισμός Άμυνας/ Ευπάθεια	**Αριθμός ιστοσελίδων (Σχετικό σύνολο)**	**Ποσοστό Ιστοσελίδων (%)**
HttpOnly	703 (1376)	51,09
SecureCookies	105 (1376)	7,63
Strict-Transport-Security	136 (2661)	5,11
X-Frame-Options	340 (2661)	12,77
X-XSS-Protection	455 (2661)	17,09
X-Content-Type-Options	491 (2661)	18,45
Content Security Policy	17 (2661)	0,638
Stale Javascript Inclusions	11 (1880)	0,585
Iframe sandboxing	1 (1027)	0,097
CSRF tokens	118 (1862)	6,337
SSL-Stripping vulerable form	161 (1862)	8,646

Είναι σημαντικό να αποσαφηνιστεί το γεγονός ότι, ορισμένα από τα αποτελέσματα που παρουσιάζονται στον παραπάνω πίνακα αφορούν ένα υποσύνολο των ιστοσελίδων που αξιολογήθηκαν. Συγκεκριμένα, το ποσοστό που αναφέρεται στους μηχανισμούς που αφορούν cookies αφορά τις ιστοσελίδες εκείνες που έχουν χρησιμοποιήσει cookies. Αντίστοιχα, η ευπάθεια *Stale Javascript Inclusions* αφορά εκείνο το υποσύνολο που έχει συμπεριλάβει κώδικα JavaScript από τρίτους, ενώ το *Iframe sandboxing* συμπεριλαμβάνει εκείνες τις ιστοσελίδες που έχουν χρησιμοποιήσει κάποιο HTML στοιχείο *iframe*. Τέλος, ο μηχανισμός άμυνας *CSRF token*, καθώς και η ευπάθεια *SSL-Stripping vulnerable form* αφορούν το σύνολο των ιστοσελίδων που περιέχουν κάποια HTML φόρμα.

Από τα πιο αξιοσημείωτα σημεία των παραπάνω αποτελεσμάτων είναι ότι περισσότερες από τις μισές ιστοσελίδες που κάνουν χρήση cookies, χρησιμοποιούν την ιδιότητα *HttpOnly*, κάτι αρκετά σημαντικό, αλλά κι ενθαρρυντικό, αφού εξασφαλίζει ότι cookies που αποθηκεύουν ευαίσθητη πληροφορία δεν μπορούν να υποκλαπούν μέσω επιθέσεων *Cross Site Scripting*. Από την άλλη, μόλις ένα ποσοστό 7,63%

ασφαλίζει τα cookies με την ιδιότητα *secure*, πράγμα που σημαίνει ότι οι υπόλοιπες ιστοσελίδες αποστέλλουν τα cookies τους και μέσω HTTP, χωρίς καμία κρυπτογράφηση, κάτι που κάνει εύκολη την υποκλοπή τους μέσω κάποιας επίθεσης *Man in The Middle*.

Σε ό,τι αφορά τους υπόλοιπους μηχανισμούς ασφάλειας που υλοποιούνται ως οδηγίες στον browser του χρήστη μέσω HTTP Headers, παρατηρούμε ότι στην πλειοψηφία τους χρησιμοποιούνται από αρκετά μικρά ποσοστά ιστοσελίδων. Την συχνότερη εμφάνιση έχει το *X-Content-Type-Options* που υλοποιείται από το 18,45% των ιστοσελίδων που αξιολογήθηκαν κι αμέσως μετά ακολουθούν το *X-XSS-Protection* με 17,09% και το *X-Frame-Options* με 12,77%. Αρκετά λιγότερες είναι όμως εκείνες οι σελίδες που απαιτούν ασφαλή HTTPS σύνδεση του χρήστη με το *Strict-Transport-Security* header, μόλις 5,11% από το σύνολο των σελίδων που αξιολογήθηκαν. Τέλος, μόνο το 0,638% των ιστοσελίδων ορίζει κάποια πολιτική ασφάλειας με το *Content-Security-Policy.*

Σχετικά με τα ευρήματα που αφορούν HTML στοιχεία της κάθε σελίδας, από τις 1027 σελίδες που κάνουν χρήση κάποιου *iframe*, μόνο μία το έχει ασφαλίσει με την χρήση της ιδιότητας *sandbox*, ενώ από το σύνολο των ιστοσελίδων που περιέχουν κάποια HTML φόρμα, μόνο το 6,337% χρησιμοποιεί κάποιο *CSRF token* για την αποτροπή επιθέσεων *Cross Site Request Forgery*.

Σε ό,τι αφορά τις ευπαθείς ιστοσελίδες που βρέθηκαν, από το σύνολο εκείνο που περιέχει κώδικα JavaScript από εξωτερικούς παρόχους, είναι αρκετά λίγες εκείνες στις οποίες δεν μπόρεσε να βρεθεί ο πάροχος σαν έγκυρη διεύθυνση. Συγκεκριμένα, το 0,585% των ιστοσελίδων που περιέχουν εξωτερικό JavaScript κώδικα, είναι ευάλωτες στην ενσωμάτωση αυθαίρετου και κακόβουλου JavaScript κώδικα από κάποιον επιτιθέμενο. Ακόμη, στο σύνολο των ιστοσελίδων που χρησιμοποιούν κάποια HTML φόρμα, το 8,646% είναι ευάλωτο σε επιθέσεις SSL-Stripping, δηλαδή κάποιος επιτιθέμενος που έχει καταφέρει να εκτελέσει μία *Man in The Middle* επίθεση έχει την δυνατότητα να αποτρέψει την περαιτέρω ασφαλή σύνδεση του χρήστη στον ιστότοπο.

Τέλος, αξίζει να σημειωθεί ότι πολλές είναι εκείνες οι ιστοσελίδες που δίνουν πληροφορίες σχετικά με την έκδοση και το λειτουργικό σύστημα του server τους, αλλά και για το λογισμικό ή την γλώσσα προγραμματισμού που χρησιμοποιούν, μέσω των HTTP headers *Server* και *X-Powered-By*. Πιο συγκεκριμένα, το 93,98% των ιστοσελίδων δίνει κάποια πληροφορία για την έκδοση του server και το 33,37% για την γλώσσα που χρησιμοποιεί. Αν και δεν αποτελούν κάποιον μηχανισμό άμυνας ή κάποια ευπάθεια, είναι αξιοσημείωτα ευρήματα διότι μπορεί να δίνουν χρήσιμη πληροφορία σε κάποιον επιτιθέμενο, π.χ. απαρχαιωμένες εκδόσεις του server ή της γλώσσας που χρησιμοποιεί με γνωστές ευπάθειες που μπορεί να τις εκμεταλλευτεί.

5.2.1 Ελληνικές ιστοσελίδες

Είναι ουσιώδες να περιγραφεί συνοπτικά και το υποσύνολο των παραπάνω αποτελεσμάτων που αφορούν αποκλειστικά ελληνικές ιστοσελίδες. Για τον σκοπό αυτόν, συλλέχθηκαν μόνο οι ιστοσελίδες που έχουν την κατάληξη *.gr*, γεγονός που απέκλεισε κάποιες ελληνικές ιστοσελίδες που μπορεί να είχαν διαφορετική κατάληξη, π.χ. *.gov*, *.edu*, *.com* κλπ. Στο σύνολό τους οι ιστοσελίδες αυτές ήταν 1182.

Πίνακας 5.4 Σκορ που αποδόθηκαν κατά την εκτέλεση σε ελληνικές ιστοσελίδες

Κατηγορία Σκορ	**Αριθμός ιστοσελίδων**	**Ποσοστό Ιστοσελίδων (%)**
Critical (<0)	13	1,099
Low (0-15)	1147	97,03
Mid (15-30)	22	1,861
High (30-47,2)	0	0

Από τα σκορ που αποδόθηκαν σε ελληνικές ιστοσελίδες, το πιο αξιοσημείωτο είναι ότι καμία δεν έχει υλοποιήσει αρκετούς μηχανισμούς ασφάλειας, ώστε να πάρει ένα υψηλό σκορ. Εξίσου σημαντικό, είναι το γεγονός ότι λίγες είναι κι εκείνες οι ιστοσελίδες που έχουν αρνητικό σκορ, δηλαδή κάποια επιβεβαιωμένη ευπάθεια.

Πίνακας 5.5 Σκορ εμπιστοσύνης που αποδόθηκαν στις αρχικές ελληνικές ιστοσελίδες

Κατηγορία Σκορ	**Αριθμός ιστοσελίδων**	**Ποσοστό Ιστοσελίδων (%)**
Critical (<0)	0	0
Low (0-15)	102	58,62
Mid (15-30)	64	36,78
High (30-47,2)	8	4,597

Σχετικά με τα σκορ εμπιστοσύνης των ελληνικών ιστοσελίδων, τα ευρήματα είναι της ίδιας τάξεως με τα αποτελέσματα που αφορούν το σκορ εμπιστοσύνης του γενικού συνόλου των ιστοσελίδων που αξιολογήθηκαν.

Πίνακας 5.6 Ευρήματα σχετικά με μηχανισμούς άμυνας κι ευπάθειες στις ελληνικές ιστοσελίδες

Μηχανισμός Άμυνας/ Ευπάθεια	**Αριθμός ιστοσελίδων (Σχετικό σύνολο)**	**Ποσοστό Ιστοσελίδων (%)**
HttpOnly	163 (433)	37,64
SecureCookies	15 (433)	3,464
Strict-Transport-Security	9 (1182)	0,761

X-Frame-Options	22 (1182)	1,861
X-XSS-Protection	300 (1182)	25,38
X-Content-Type-Options	297 (1182)	25,12
Content Security Policy	2 (1182)	0,169
Stale Javascript Inclusions	9 (747)	1,204
Iframe sandboxing	1 (512)	0,195
CSRF tokens	37 (737)	5,02
SSL-Stripping vulerable form	99 (737)	13,43

Συγκριτικά με τα γενικά αποτελέσματα, οι ελληνικές ιστοσελίδες στο σύνολό τους φαίνεται να υστερούν στους μηχανισμούς άμυνας, με εξαίρεση τα HTTP Headers *X-XSS-Protection* και *X-Content-Type-Options* που έχουν καλύτερα ποσοστά εμφάνισης. Ακόμη, συνολικά φαίνεται να έχουν περισσότερα ποσοστά ευπαθειών *Stale Javascript Inclusions* και *SSL-Stripping*. Τέλος, μεγάλα είναι τα ποσοστά των ελληνικών ιστοσελίδων που φανερώνουν πληροφορίες για τον server και την γλώσσα προγραμματισμού ή λογισμικό που χρησιμοποιεί, συγκεκριμένα 95,6% και 36,04% αντίστοιχα.

5.3 Συμπεράσματα

Με βάση τα αποτελέσματα που παρουσιάστηκαν παραπάνω κι έχοντας πάντα υπόψιν τον προσεγγιστικό τους χαρακτήρα, μπορούμε να εξάγουμε κάποια κοινά συμπεράσματα για δύο ευρείες, αλληλένδετες κατηγορίες: την ασφαλή περιήγηση των Ελλήνων χρηστών στο διαδίκτυο, αλλά και την μέριμνα που λαμβάνεται από τους ιδιοκτήτες ελληνικών ιστοσελίδων για την ασφάλειά των ίδιων των σελίδων αλλά και των χρηστών τους.

Αρχικά, μπορούμε να παρατηρήσουμε ότι οι ιστοσελίδες που επισκέπτονται οι Έλληνες χρήστες του διαδικτύου, άρα και οι ελληνικές ιστοσελίδες, έχουν αρκετά περιθώρια βελτίωσης της ασφάλειάς τους, σε ό,τι αφορά τα χαρακτηριστικά που εξετάζει η παρούσα εργασία. Συγκεκριμένα, μηχανισμοί άμυνας, είτε στο επίπεδο των HTTP headers, είτε στο επίπεδο του HTML κώδικα, που δεν επηρεάζουν την λειτουργικότητα μιας ιστοσελίδας, ή την επηρεάζουν ελάχιστα με κέρδος όμως την καλύτερη ασφάλειά της, θα μπορούσαν και θα έπρεπε να είναι ενεργοί, για παράδειγμα το header *Strict-Transport-Security* ή το *Content-Security-Policy*. Ακόμη ένα καλό παράδειγμα, αποτελεί η σχεδόν παντελής έλλειψη της ιδιότητας *sandbox* από τα HTML iframes, δίνοντας το ελεύθερο σε επιτιθέμενους που καταφέρνουν να φορτώσουν μία δική τους, κακόβουλη ιστοσελίδα στο iframe να εκτελούν αυθαίρετα ότι κώδικα θέλουν. Κάτι ακόμα που φανερώνει την ανάγκη για βελτίωση της ασφάλειας ορισμένων ιστοσελίδων και την έλλειψη μέριμνας

αυτής, είναι ότι ένα σχετικά καλό ποσοστό του συνόλου, περίπου μία στις 10, είναι ευάλωτο σε επιθέσεις SSL-Stripping.

Από την άλλη πλευρά, είναι ενθαρρυντικό το ότι πολλές από τις ιστοσελίδες χρησιμοποιούν *HttpOnly cookies*, γεγονός που δείχνει ότι οι ιδιοκτήτες τους έχουν λάβει την ασφάλειά τους, εν μέρει, υπόψιν. Ακόμη, αρκετά αισιόδοξο είναι το ότι οι ευπαθείς ιστοσελίδες σε εκτέλεση κακόβουλου JavaScript κώδικα τρίτων, που βρέθηκαν, αποτελούν ένα αρκετά μικρό υποσύνολο, σχεδόν μηδενικό.

Εν κατακλείδι, θα μπορούσε κανείς να πει ότι κατά κανόνα οι ιδιοκτήτες των ελληνικών ιστοσελίδων, κατά την δημιουργία και την συντήρησή τους δεν έχουν την ασφάλειά τους κατά νου ή δεν της δίνουν την βαρύτητα που απαιτείται. Κατά συνέπεια, μπορεί επίσης να ειπωθεί με κάθε επιφύλαξη, ότι οι Έλληνες περιηγητές του διαδικτύου δεν είναι τόσο ασφαλείς κατά την περιήγησή τους και τις δραστηριότητές τους σε αυτό, όσο θα ήταν εφικτό κι όσο θα έπρεπε.

Κεφάλαιο 6

Συμπεράσματα

6.1 Συμπεράσματα

Στην παρούσα πτυχιακή εργασία αναπτύχθηκε μία εφαρμογή για ηλεκτρονικούς υπολογιστές, με την χρήση της γλώσσας προγραμματισμού Python, με στόχο την συλλογή δημόσια διαθέσιμων πληροφοριών συσχετισμένων με την ασφάλεια από ιστοσελίδες, την αξιολόγηση τους, την παρουσίαση των ευρημάτων της εκτέλεσης της εφαρμογής και με απώτερο στόχο να δίνει την δυνατότητα στον χρήστη να εξάγει συμπεράσματα και γνώση για μεμονωμένες ιστοσελίδες ή για σύνολα αυτών.

Η εφαρμογή προκειμένου να μπορεί να εκτελεστεί σε ένα μεγάλο εύρος ιστοσελίδων σχεδιάστηκε έτσι ώστε να λειτουργεί και σαν web crawler, αλλά και web scraper, όταν το ορίσει ο χρήστης. Η λειτουργικότητα όμως αυτή της εφαρμογής έθεσε και κάποια άλλα βασικά ζητήματα που έπρεπε να ληφθούν υπόψιν. Βασικότερο ήταν η προσοχή στην συγγραφή του κώδικα ώστε να μην γίνονται περιττά βήματα κι επαναλήψεις στην εκτέλεση, ώστε να διασφαλίζεται κατά το δυνατό η ταχύτητα του crawler. Σε συνδυασμό με αυτό, η παράκαμψη ιστοσελίδων που είχαν αξιολογηθεί ξανά κατά την εκτέλεση, πέρα από την βελτίωση του χρόνου εκτέλεσης, συνέβαλλε και στην εξάλειψη διπλότυπων αποτελεσμάτων, ένα ακόμα ζήτημα που έπρεπε να αντιμετωπιστεί. Ακόμη, εξίσου βασικό, για να διασφαλιστεί και η συνεχής κι αδιάκοπη εκτέλεση της εφαρμογής, ειδικά όταν πρόκειται για ένα crawling μερικών χιλιάδων ιστοσελίδων, μιας διαδικασίας που απαιτεί κάποιον χρόνο, έπρεπε να ληφθούν υπόψιν και τα πιθανά σφάλματα που μπορεί να προκύψουν κατά την εκτέλεση και κατ'επέκταση και η διαχείρισή τους.

Η προσέγγιση της παθητικής ανάλυσης που ακολουθεί η εφαρμογή, προκειμένου να μην παραβιάζεται κανένας ηθικός και νομικός θεσμός, ήταν μία δέσμευση ως προς το εύρος των χαρακτηριστικών που συλλέγει κι αξιολογεί. Συγκεκριμένα, τα χαρακτηριστικά τα οποία συλλέγονται από κάθε ιστοσελίδα χωρίζονται σε δύο κύριες κατηγορίες, τα HTTP Headers και τα στοιχεία HTML. Ενώ

πράγματι όλα αυτά τα χαρακτηριστικά στοιχεία σχετίζονται με την ασφάλεια ενός ιστοτόπου, δεν αποτελούν τις μόνες παραμέτρους που την επηρεάζουν και την χαρακτηρίζουν. Σε μία πιο ενεργητική προσέγγιση θα μπορούσαν να ελεγχθούν περισσότερες ευπάθειες, κάτι που θα διασφάλιζε μία αντικειμενικότερη εικόνα για την ασφάλεια ενός ιστοτόπου, αλλά θα απαιτούσε και την συγκατάθεση κι έγκριση του ιδιοκτήτη του, καθώς επίσης θα ήταν δεσμευτικό και ως προς τον χρόνο εκτέλεσης.

Βασικό συστατικό στοιχείο της εφαρμογής είναι και η μετρική αξιολόγησης που χρησιμοποιήθηκε. Προκειμένου να αναπτυχθεί μία σχετικά αντικειμενική κι αντιπροσωπευτική μετρική έπρεπε να επιλεχθεί ένα υπάρχον σύστημα που έχει χρησιμοποιηθεί στο παρελθόν και να προσαρμοστεί ανάλογα. Για αυτόν τον λόγο χρησιμοποιήθηκε το Common Vulnerability Scoring System v 3.0 που έχει αναπτυχθεί από το FIRST. [2] Το σύστημα αυτό όμως, όπως φαίνεται κι από το όνομά του, αφορά ευπάθειες αποκλειστικά κι όχι μηχανισμούς άμυνας. Επομένως, για να χρησιμοποιηθεί από την εφαρμογή, σε κάθε μηχανισμό άμυνας που ελέγχεται ανατέθηκε το σκορ της αντίστοιχης ευπάθειας που αποτρέπει. Επίσης, το σκορ που αποδίδεται σε κάθε μηχανισμό άμυνας ή ευπάθεια υπολογίζεται από τον αναλυτή, επιλέγοντας συγκεκριμένα χαρακτηριστικά της ευπάθειας. Έτσι, τα σκορ που αποδόθηκαν είναι προσεγγιστικά και κατά το δυνατόν ουδέτερα ώστε να αντικατοπτρίζουν μία γενική μορφή της εκάστοτε ευπάθειας. Για τους παραπάνω λόγους, τα αποτελέσματα που παράγονται από την εφαρμογή είναι προσεγγιστικά κι όχι απόλυτα.

Παρά ταύτα, η εφαρμογή που αναπτύχθηκε στα πλαίσια της παρούσας πτυχιακής εργασίας ανταποκρίνεται στους στόχους της, αφού καταφέρνει να αξιολογήσει αυτοματοποιημένα ένα αρκετά μεγάλο σύνολο ιστοσελίδων σε ικανοποιητικό χρόνο και να εξάγει χρήσιμα συμπεράσματα για το σύνολο αυτό. Επίσης, η καταγραφή των αποτελεσμάτων δίνει την δυνατότητα να μελετηθούν περαιτέρω κι από άλλους χρήστες και μάλιστα, λόγω της μορφοποίησής τους, ανεξάρτητα από την γλώσσα προγραμματισμού που χρησιμοποιείται.

Εν τέλει, η εφαρμογή που αναπτύχθηκε μπορεί να χρησιμοποιηθεί από απλούς χρήστες για τον έλεγχο της δικής τους ή άλλων ιστοσελίδων, αλλά και από πιο εξειδικευμένους χρήστες, π.χ. αναλυτές ασφάλειας, για την εξαγωγή συμπερασμάτων και γνώσης για ένα εύρος ιστοσελίδων που τους ενδιαφέρει. Παράλληλα, η πληροφορία που εξάγει η εφαρμογή μπορεί εύκολα να επεκταθεί με την προσθήκη ελέγχου άλλων ευπαθειών και μηχανισμών άμυνας.

6.2 Μελλοντικές επεκτάσεις

Η ασφάλεια των πληροφοριακών συστημάτων και ιδιαίτερα σε ό,τι αφορά το διαδίκτυο, αποτελεί έναν τομέα με ιδιαίτερο ενδιαφέρον που έχει μελετηθεί αρκετά και που θα συνεχίσει να απασχολεί και να αφορά τόσο τους ειδικούς του χώρου, όσο και τους απλούς χρήστες. Επίσης, είναι ένας τομέας που θα συνεχίσει να εξελίσσεται αναπόφευκτα μαζί με κάθε άλλη πτυχή της τεχνολογίας. Αυτά τα χαρακτηριστικά

και μόνο καθιστούν τα περιθώρια επέκτασης της εφαρμογής αρκετά και μεγάλου εύρους.

Αρχικά, αρκετά χρήσιμη θα ήταν η μεταφορά της εφαρμογής και στις νεότερες εκδόσεις της γλώσσας Python, έτσι ώστε να συνεχίζει να υποστηρίζεται μακροπρόθεσμα η λειτουργία της και η βελτίωσή της. Παράλληλα, με την χρήση εργαλείων, όπως το Kivy [18], θα μπορούσε να μεταφερθεί και σε άλλες πλατφόρμες, όπως το λειτουργικό σύστημα για κινητές συσκευές Android.

Επίσης, μία χρήσιμη βελτίωση της εφαρμογής θα ήταν η βελτιστοποίηση ή η προσαρμογή του αλγορίθμου του crawling, με στόχο την καλύτερη απόδοση της εφαρμογής. Για παράδειγμα, θα μπορούσε να γίνεται συγκεντρωμένη αποθήκευση των ιστοσελίδων που έχουν αξιολογηθεί σε κάποια κεντρική βάση δεδομένων ή αποθετήριο και με αποδοτικές μεθόδους προσπέλασης να παρακάμπτονται κατά την εκτέλεση αυτές που έχουν αξιολογηθεί πρόσφατα. Έτσι, ανεξάρτητα με το ποιος χρήστης χρησιμοποιεί την εφαρμογή θα μπορεί να έχει αποδοτικότερη εκτέλεση και ταυτόχρονα να συμβάλλει στην βελτίωση της εκτέλεσης άλλων χρηστών.

Ακόμη, θα ήταν πολύ χρήσιμο αν υλοποιούνταν ο έλεγχος ήδη γνωστών, σημαντικών ευπαθειών, για παράδειγμα του γνωστού Heartbleed ή ο έλεγχος για SQL injection ή Cross Site Scripting επιθέσεις που είναι πολύ διαδεδομένες. Βέβαια, οι έλεγχοι αυτοί θα εκτελούνταν μετά από σχετική παράμετρο που θα όριζε ο χρήστης πριν την εκτέλεση της εφαρμογής.

Κατ'επέκταση και σε γενικότερα πλαίσια, η εφαρμογή θα μπορούσε να ακολουθεί οποιαδήποτε νέα πρακτική εξέλιξη στο χώρο της διαδικτυακής ασφάλειας, όπως τη δημοσιοποίηση καινούργιων ευπαθειών ή μηχανισμών άμυνας. Με τον τρόπο αυτό ο χρήστης θα μπορούσε, εν γνώση του και με δική του ευθύνη, να ακολουθεί μία πιο ενεργητική προσέγγιση για τον έλεγχο των ιστοσελίδων που τον ενδιαφέρουν και άρα να ανακτά χρησιμότερα και πιο αντικειμενικά αποτελέσματα. Οι πιθανές αυτές επεκτάσεις της εφαρμογής όμως, καθιστούν ακόμα σημαντικότερη την επέκτασή της σε ό,τι αφορά την απόδοσή της και τον χρόνο εκτέλεσης, αφού όσο περισσότεροι οι έλεγχοι, τόσος περισσότερος ο χρόνος που απαιτείται. Ακόμη, η εφαρμογή θα πρέπει να ανανεώνεται σε ό,τι αφορά, εκτός από νέες ευπάθειες κλπ, την κατάργηση μηχανισμών άμυνας ή την καθολική εξάλειψη ευπαθειών που δεν έχει νόημα να ελέγχονται πλέον.

Τέλος, θα μπορούσε να υλοποιηθεί μία απλή διεπιφάνεια χρήστη, έτσι ώστε το περιβάλλον της εφαρμογής να είναι πιο φιλικό και κατανοητό προς τον χρήστη και να μπορεί να χρησιμοποιηθεί κι από άτομα που είναι πιο εξοικειωμένα με το γραφικό περιβάλλον εφαρμογών κι όχι με την γραμμή εντολών.

Βιβλιογραφία

[1] OWASP. List of useful HTTP headers. https://www.owasp.org/index.php/List_of_useful_HTTP_headers [Online; τελευταία προσπέλαση 9-Αυγούστου-2015].

[2] FIRST. Common Vulnerability Scoring System version 3.0 Calculator. https://www.first.org/cvss/calculator/3.0 [Online; τελευταία προσπέλαση 9-Αυγούστου-2015].

[3] Wikipedia, The Free Encyclopedia. Information Security. https://en.wikipedia.org/wiki/Information_security#History [Online; τελευταία προσπέλαση 12-Αυγούστου-2015].

[4] Wikipedia, The Free Encyclopedia. Information Security. https://en.wikipedia.org/wiki/Information_security#Basic_principles [Online; τελευταία προσπέλαση 12-Αυγούστου-2015].

[5] Wikipedia, The Free Encyclopedia. Threat (Computer). https://en.wikipedia.org/wiki/Threat_(computer)#Threat_classification [Online; τελευταία προσπέλαση 12-Αυγούστου-2015].

[6] tom's GUIDE. 10 Worst Data Breaches of All Time http://www.tomsguide.com/us/biggest-data-breaches,news-19083.html [Online; τελευταία προσπέλαση 12-Αυγούστου-2015].

[7] Wikipedia, The Free Encyclopedia. Web crawler. https://en.wikipedia.org/wiki/Web_crawler#Crawling_policy [Online; τελευταία προσπέλαση 13-Αυγούστου-2015].

[8] Wikipedia, The Free Encyclopedia. Web crawler. https://en.wikipedia.org/wiki/Web_crawler#Crawler_identification [Online; τελευταία προσπέλαση 13-Αυγούστου-2015].

[9] Wikipedia, The Free Encyclopedia. Web crawler. https://en.wikipedia.org/wiki/Web_crawler#Security [Online; τελευταία προσπέλαση 13-Αυγούστου-2015].

[10] Python https://www.python.org/ [Online; τελευταία προσπέλαση 17-Αυγούστου-2015].

[11] Python https://www.python.org/about [Online; τελευταία προσπέλαση 17-Αυγούστου-2015].

[12] Wikipedia, The Free Encyclopedia. History of Python. https://en.wikipedia.org/wiki/History_of_Python [Online; τελευταία προσπέλαση 17-Αυγούστου-2015].

[13] Nick Nikiforakis, Luca Invernizzi, Alexandros Kapravelos, Steven Van Acker, Wouter Joosen, Christopher Kruegel, Frank Piessens, and Giovanni Vigna. You Are What You Include: Large-scale Evaluation of Remote JavaScript Inclusions. In Proceedings of the 2012 ACM conference on Computer and communications security, CCS '12, pages 736–747, New York, NY, USA, 2012. ACM.

[14] OWASP. Content Security Policy. https://www.owasp.org/index.php/Content_Security_Policy [Online; τελευταία προσπέλαση 13-Αυγούστου-2015].

[15] JSON. http://json.org/json-el.html [Online; τελευταία προσπέλαση 20-Αυγούστου-2015].

[16] Wikipedia, The Free Encyclopedia. Depth First Search https://en.wikipedia.org/wiki/Depth-first_search [Online; τελευταία προσπέλαση 6-Σεπτέμβρη-2015]

[17] Alexa. Top Sites in Greece. http://www.alexa.com/topsites/countries/GR [Online; τελευταία προσπέλαση 26-Αυγούστου-2015].

[18] Kivy - Open source Python library for rapid development of applications that make use of innovative user interfaces, such as multi-touch apps. http://kivy.org/ [Online; τελευταία προσπέλαση 27-Αυγούστου-2015].