Πανεπιστήμιο Πειραιώς – Τμήμα Πληροφορικής

Πρόγραμμα Μεταπτυχιακών Σπουδών
«Προηγμένα Συστήματα Πληροφορικής»

Μεταπτυχιακή Διατριβή

Τίτλος Διατριβής	Web Ninja – Εργαλείο ανάλυσης ύποπτων αρχείων JavaScript σε πραγματικό χρόνο «Web Ninja – A real time malicious JavaScript analysis tool»
Ονοματεπώνυμο Φοιτητή	Ρουσογιαννάκης Λιάγκος Σπυρίδων
Πατρώνυμο	Ευτύχιος
Αριθμός Μητρώου	ΜΠΣΠ 13094
Επιβλέπων	Πατσάκης Κωνσταντίνος, Επίκουρος καθηγητής

Ημερομηνία Παράδοσης: Οκτώβριος 2016

Η παρούσα Μεταπτυχιακή Διατριβή εγκρίθηκε ομόφωνα από την Τριμελή Εξεταστική Επιτροπή που ορίσθηκε από τη ΓΣΕΣ του Τμήματος Πληροφορικής του Πανεπιστημίου Πειραιώς σύμφωνα με τον Εσωτερικό Κανονισμό Λειτουργίας του Προγράμματος Μεταπτυχιακών Σπουδών “Προηγμένα Συστήματα Πληροφορικής”.

Τριμελής Εξεταστική Επιτροπή

(υπογραφή)	(υπογραφή)	(υπογραφή)
Κωνσταντίνος Πατσάκης Επίκουρος καθηγητής	Ευθύμιος Αλέπης Επίκουρος καθηγητής	Γεώργιος Τσιχριντζής Καθηγητής

Μεταπτυχιακή Διατριβή

«Web Ninja – Εργαλείο ανάλυσης ύποπτων αρχείων JavaScript σε διαδικτυακούς ιστότοπους σε πραγματικό χρόνο»

Επιμέλεια:

Ρουσογιαννάκης Λιάγκος Σπυρίδων

Ευχαριστίες

Η παρούσα διπλωματική διατριβή υλοποιήθηκε στα πλαίσια του Προγράμματος Μεταπτυχιακών Σπουδών «Προηγμένα Συστήματα Πληροφορικής» του Πανεπιστημίου Πειραιώς.

Θα ήθελα να ευχαριστήσω τον καθηγητή μου Κο Πατσάκη Κωνσταντίνο για την πολύτιμη βοήθειά του κατά την διάρκεια όλης της φοίτησής μου στο τμήμα. Περεταίρω θα ήθελα να ευχαριστήσω την οικογένειά μου, που με στηρίζει όλα αυτά τα χρόνια στην κάθε μου επιλογή.

Ιδιαίτερο και μεγάλο ευχαριστώ θα ήθελα να πω στους υπόλοιπους που με στήριξαν και με βοήθησαν αυτά τα χρόνια και ιδιαίτερα τους τελευταίους μήνες στο να ανταπεξέλθω στις δυσκολίες και τα εμπόδια και να ολοκληρώσω με αυτή την εργασία το μεταπτυχιακό αυτό πρόγραμμα. Ιδιαίτερο ευχαριστώ λοιπόν στο Δεσποινάκι που είναι δίπλα μου χρόνια τώρα και μου δίνει δύναμη και κουράγιο, στον Μήτσο που με πιέζει πάντα να βγάζω τον καλύτερό μου εαυτό και να προχωράω μπροστά. Μεγάλο ευχαριστώ στον Νικόλα που χωρίς την βοήθεια και την υποστήριξή του δεν θα τα είχα καταφέρει να ολοκληρώσω αυτή την εργασία.

Οκτώβριος 2016
Ρουσογιαννάκης Λιάγκος Σπυρίδων

Περίληψη

Σκοπός αυτής της διπλωματικής εργασίας είναι η μελέτη, ανάλυση και η δημιουργία ενός εργαλείου ελέγχου και ενημέρωσης του χρήστη, για την επικινδυνότητα της ιστοσελίδας την οποία επισκέπτεται. Η ανάλυση του κώδικα γίνεται σε πραγματικό χρόνο με σκοπό την αποφυγή μόλυνσης του υπολογιστή του εκάστοτε χρήστη. Για την ανάπτυξη έγινε χρήση των Chrome platform APIs με παράλληλη χρήση του Node.js για το server side κομμάτι.

Abstract

The purpose of this dissertation is the study, the analysis and the creation of a tool to protect the user from malicious website he might visit. The tool will analyze at real time and re-write the code of the website in such a way to protect the system and personal data of the user. In order to develop such an application, the Chrome platform APIs and Node.js were used.

Πίνακας περιεχομένων

1. Πρόλογος

Το διαδίκτυο είναι η μεγαλύτερη τεχνολογική επανάσταση των τελευταίων δεκαετιών. Αποτελεί τη μεγαλύτερη πηγή πληροφοριών και γνωρίζει μεγάλη απήχηση από τους χρήστες σε όλο το κόσμο. Ζούμε σε ένα συνδεδεμένο κόσμο που έχει τη δυνατότητα να αναζητά πληροφορίες μέσω του ηλεκτρονικού του υπολογιστή ή της φορητής συσκευής που χρησιμοποιεί (smartphone, tablet) ανά πάσα στιγμή. Το διαδίκτυο χρησιμοποιείται από κάθε λογής χρήστες ακόμη και από μικρά παιδιά με αποτέλεσμα πολλές φορές να τίθενται θέματα ασφάλειας και διαρροής των ευαίσθητων προσωπικών δεδομένων των χρηστών. Το διαδίκτυο αποτελεί μέρος της ζωής μας και η ανάγκη του να προστατευτούμε από τις κακόβουλες επιθέσεις μεγαλώνει ολοένα και περισσότερο.

Πολλές φορές οι χρήστες δεν προσέχουν τις κινήσεις τους κατά τη διαδικασία μιας απλής περιήγησης και βρίσκονται εκτεθειμένοι σε κακόβουλο λογισμικό. Σκοπός αυτής της διπλωματικής εργασίας είναι η μελέτη, σχεδίαση και ανάπτυξη ενός εργαλείου που θα βρίσκει σε πραγματικό χρόνο και θα διορθώνει τμήματα μολυσμένου κώδικα (Javascript) που χρησιμοποιείται στις ιστοσελίδες που περιηγείται ο χρήστης και μπορούν να επιφέρουν βλάβη στον προσωπικό υπολογιστή και κατ' επέκταση στα προσωπικά του αρχεία, προστατεύοντας τον ίδιο το χρήστη και τα προσωπικά του δεδομένα.

2. Εισαγωγή

Με την συνεχή εξέλιξη της τεχνολογίας και την ανάπτυξη των OnLine υπηρεσιών το μεγαλύτερο κομμάτι των χρηστών του διαδικτύου αποτελείται πλέον από απλούς καθημερινούς ανθρώπους οι οποίοι χρησιμοποιούν την τεχνολογία όλο και σε μεγαλύτερο βαθμό στην καθημερινότητα τους. Είναι πλέον επιτακτική ανάγκη για τους πολίτες η σύνδεσή τους στο διαδίκτυο όμως όχι και η πιο χαλαρή προστασία των προσωπικών τους δεδομένων. Πολλές φορές οι χρήστες δεν γνωρίζουν τους κινδύνους και το τι μπορεί να συμβεί σε περίπτωση κακόβουλου λογισμικού ή σε περίπτωση που προσβληθούν από κακόβουλο κώδικα κατά τη διάρκεια της περιήγησης τους στο διαδίκτυο. Οι συνέπειες μπορεί να έχουν τραγικά αποτελέσματα καθώς θίγονται ευαίσθητα προσωπικά δεδομένα αλλά και πληροφορίες οι οποίες δε θα έπρεπε ποτέ να δημοσιοποιηθούν ή να διαρρεύσουν.

Ζούμε στην εποχή του Internet of Things (IoT) με όλο και περισσότερο κόσμο να στρέφεται στη τεχνολογία και να δέχεται να παραμένει συνδεδεμένος στο διαδίκτυο σχεδόν το 90% της ημέρας. Πλέον οι περισσότερες ή σχεδόν όλες οι φορητές συσκευές που χρησιμοποιούμε είναι συνδεδεμένες στο διαδίκτυο, προσφέροντας ασταμάτητη συλλογή δεδομένων. Η ασταμάτητη συλλογή των δεδομένων από τις συσκευές αυτές δημιουργεί την ανάγκη για μεγαλύτερη ασφάλεια. Τα δεδομένα αυτά μπορεί μεν να είναι απλοί αριθμοί όμως στην πραγματικότητα είναι κατά κόρον ευαίσθητα προσωπικά δεδομένα που μπορούν να αποκαλύψουν σε άτομα με την κατάλληλη τεχνογνωσία τις συνήθειες, τα ενδιαφέροντα, τα μέρη στα οποία κινείται ένας χρήστης, τις επιλογές του, τις προτιμήσεις του ακόμη και τις συζητήσεις του.

Οι άνθρωποι σήμερα είναι σε θέση μέσω της σύγχρονης τεχνολογίας και της υψηλής συνδεσιμότητας να επικοινωνούν με οποιοδήποτε μέρος του πλανήτη, να πραγματοποιούν ηλεκτρονικές αγορές μέσα από το smartphone τους, να χρησιμοποιούν το διαδίκτυο σε όλο και περισσότερους τομείς, όπως για παράδειγμα στην εκπαίδευση, να εργαστούν εξ' αποστάσεως και να πραγματοποιούν συναλλαγές με διάφορες υπηρεσίες που προσφέρονται από τις τράπεζες με σκοπό την διευκόλυνση του αγοραστικού κοινού. Με αυτό το τρόπο οι χρήστες είναι όλο και περισσότερο εκτεθειμένοι στον κόσμο του διαδικτύου.

Οι επαγγελματίες που ακολουθούν τη τάση της αγοράς θεωρούν ότι με τη χρήση της τεχνολογίας γίνονται πιο ανταγωνιστικοί και ότι τους βοηθάει η τεχνολογία στο να κερδίσουν μεγαλύτερο κομμάτι στην ήδη ανταγωνιστική αγορά. Αυτό ισχύει και είναι προφανές πως η τεχνολογία μας βοηθάει στο να γινόμαστε καλύτεροι, όμως η έκθεσή των προσωπικών μας δεδομένων στο διαδίκτυο χωρίς να λαμβάνονται τα κατάλληλα μέτρα προστασίας αυτών είναι κάτι που πολλές φορές μπορεί να αποβεί μοιραίο. Ο επαγγελματίας και ο επιστήμονας το θεωρεί ένα από τα χρησιμότερα εργαλεία για την δουλειά του και αφιερώνει πολλές ώρες σε αυτό. Αυτό όμως θα μπορούσε να είναι ένα σημάδι εξάρτησης.

Η ιστορία του διαδικτύου ξεκινάει πολλές δεκαετίες πίσω, την περίοδο του ψυχρού πολέμου. Η ανάγκη των ΗΠΑ στο να προστατευτούν από μία ενδεχόμενη επίθεση της Ρωσίας με χρήση πυρηνικού οπλισμού, τους έκανε να συστήσουν μία υπηρεσία προηγμένων αμυντικών ερευνών. Την ARPA (Advanced Research Project Agency) η οποία έγινε γνωστή και ως DARPA (Defense Advanced Research Projects Agency). Η συγκεκριμένη υπηρεσία είχε ως αποστολή το να βοηθήσει με κάθε τρόπο τις στρατιωτικές δυνάμεις των ΗΠΑ εναντίον του εχθρού, παρέχοντας τεχνολογική ανάπτυξη με σκοπό τη δημιουργία ενός τηλεπικοινωνιακού δικτύου το οποίο θα μπορούσε να επιβιώσει μετά από μία ενδεχόμενη πυρηνική επίθεση.

Πρώιμες αναφορές στην απόπειρα των ΗΠΑ αναφέρουν ότι το δίκτυο στην αρχή είχε την ονομασία «Γαλαξιακό Δίκτυο» και στην ουσία ήταν ένα δίκτυο από υπολογιστές που ήταν μεταξύ τους συνδεδεμένοι και μπορούσαν να ανταλλάσσουν πολύ γρήγορα, για τα δεδομένα της εποχής, πληροφορίες και προγράμματα. Το δίκτυο που είχαν σκεφτεί να υλοποιήσουν είχε διάφορα corner cases που θα έπρεπε να υλοποιηθούν ώστε να παραμείνει λειτουργικό σε κάθε περίπτωση. Για αυτό το λόγο το δίκτυο θα έπρεπε να είναι αποκεντρωμένο και να λειτουργούν ανεξάρτητα μεταξύ τους οι υπολογιστές έτσι ώστε σε περίπτωση βομβαρδισμού και καταστροφής του ενός υπολογιστή οι υπόλοιποι να είναι σε θέση να συνεχίσουν την επικοινωνία μεταξύ τους. Έτσι δημιουργήθηκε το 1969 το ARPANET που ήταν στην ουσία το πρώτο είδος διαδικτύου και βασίστηκε στο σχεδιασμό ενός κατανεμημένου δικτύου που έκανε χρήση της θεωρίας ανταλλαγής πακέτων. Τα πακέτα αυτά ήταν πακέτα πληροφορίας τα οποία περιείχαν την προέλευση και τον προορισμό τους και στέλνονταν από τον έναν υπολογιστή σε κάποιον άλλο. Το ARPANET ήταν στην ουσία τέσσερις μίνι υπολογιστές συνδεδεμένοι μεταξύ τους και συνέδεαν το πανεπιστήμιο της Καλιφόρνια στη Σάντα Μπάρμπαρα, το πανεπιστήμιο της Καλιφόρνια στο Λος Άντζελες, το SRI στο Στάνφορντ και το πανεπιστήμιο της Γιούτα με ταχύτητες που έφτασαν τα 50kbps μέσω dial up σύνδεσης κάνοντας χρήση του τηλεφωνικού δικτύου. Τα επόμενα χρόνια μέχρι το 1972 οι συνδεδεμένοι υπολογιστές στο δίκτυο ARPANET είχαν φτάσει τους 23 και σιγά σιγά ξεκινάει να εφαρμόζεται το σύστημα ηλεκτρονικού ταχυδρομείο, το γνωστό σε όλους μας e-mail.

Έχοντας δημιουργήσει το πρώτο δίκτυο οι επιστήμονες δεν έμειναν με σταυρωμένα χέρια και εκείνη την εποχή δημιουργήθηκαν κι άλλα δίκτυα που χρησιμοποιούσαν διαφορετικά πρωτόκολλα (x.25, UUCP κ.α.). Αυτά τα δίκτυα ήταν συνδεδεμένα με το ARPANET. Η χρήση διαφορετικών πρωτοκόλλων και οι περιορισμοί που είχε το καθένα, οδήγησε στην ανάγκη για δημιουργία ενός πρωτοκόλλου που θα ένωνε όλα τα δίκτυα που είχαν δημιουργηθεί μέχρι τότε και δεν θα είχε περιορισμούς. Το αρχικό πρωτόκολλο που χρησιμοποιούσε αρχικά το ARPANET ήταν το Network Control Protocol γνωστό και ως NCP. Το 1974 όμως δημιουργήθηκε το γνωστό σε όλους μας TCP (Transmission Control Protocol) που λίγα χρόνια μετά εξελίχθηκε σε TCP/IP και από το 1983 και μετά ήταν το βασικό πρωτόκολλο που χρησιμοποιούσε το ARPANET. Ένα χρόνο αργότερα υλοποιήθηκε το πρώτο Domain Name System (DNS), το οποίο είναι ένα σύστημα καταγραφής 1000 κεντρικών κόμβων και οι υπολογιστές του δικτύου πλέον αναγνωρίζονται μέσα από

κωδικοποιημένες διευθύνσεις που απεικονίζονται με τη μορφή αριθμών. Στηρίζοντας τη τεράστια προσπάθεια εδραίωσης και εξέλιξης του ARPANET, το Εθνικό Ίδρυμα Επιστημών των Ηνωμένων Πολιτειών Αμερικής, το NSF, δημιούργησε το 1986 την πρώτη διαδικτυακή πανεπιστημιακή ραχοκοκαλιά. Αυτό ήταν το NSFnet. Αργότερα έγινε η προσθήκη κι άλλων σημαντικών δικτύων της εποχής. Αυτά ήταν το UseNet, FidoNet και το BitNet. Η αρχή για έναν συνδεδεμένο κόσμο είχε γίνει όμως δεν ήταν ακόμη γνωστό στον κόσμο. Το 1989 στο Ερευνητικό Ίδρυμα CERN, έγινε εφαρμογή της υπηρεσίας του διαδικτύου και σταδιακά το διαδίκτυο απέκτησε τη σημερινή του μορφή.

Το διαδίκτυο από τις πρώτες ημέρες της δημιουργίας του μέχρι σήμερα δεν έχει σταματήσει να εξελίσσεται. Θεωρείται ως η μεγαλύτερη τεχνολογική δημιουργία των τελευταίων δεκαετιών η οποία έχει καταφέρει να εισβάλει στις ζωές των ανθρώπων. Όμως στη πάροδο του χρόνου δημιουργήθηκαν πολλές προσθήκες, βελτιώσεις που έκαναν το διαδίκτυο ακόμη καλύτερο, πιο γρήγορο και πιο ευέλικτο. Αυτό το σημείο καμπής, έμελλε να αλλάξει τον τρόπο χρήσης του διαδικτύου μέχρι και σήμερα καθώς η ραγδαία ανάπτυξή του και οι συνεχόμενες προσθήκες σε αυτό, εφαρμογές, ιστοσελίδες κλπ. οδήγησε στη νέα γενιά του διαδικτύου. Για αυτό το λόγο υπήρξε η ανάγκη διαφοροποίησης του ονόματος του και εισάχθηκε για πρώτη φορά ο όρος Web 2.0. Με αυτό τον όρο, περιγράφουμε την νέα γενιά του Παγκόσμιου Ιστού που είναι διαθέσιμη σε όλο τον κόσμο ώστε να μπορούν να συνεργάζονται online και να μοιράζονται πληροφορίες για σχεδόν τα πάντα.

3. Ανασκόπηση πεδίου

Αντίστοιχες εφαρμογές (εναλλακτικές) υπάρχουν. Καμία όμως δε διατίθεται δωρεάν και ταυτόχρονα απευθύνεται σε τελικούς χρήστες είτε για οικιακή είτε για προσωπική χρήση. Οι περισσότερες από αυτές τις εφαρμογές, πρόκειται για corporate environment υπηρεσίες, με μηνιαία συνδρομή, που παρακολουθούν τη σελίδα της εταιρείας ή το δίκτυο, ελέγχοντας την περίπτωση πιθανής εισαγωγής κακόβουλου λογισμικού και μόλυνσης σε αυτά. Για την οικιακή χρήση, υπάρχουν σαφώς οι κλασσικές λύσεις internet-security. Πρόκειται για ολοκληρωμένες λύσεις με σκοπό την προστασία του υπολογιστή και των δεδομένων του χρήστη. Αν και αυτές οι λύσεις είναι πάρα πολύ καλές και σίγουρα είναι καλό κάποιος να έχει εγκατεστημένη κάποια στον προσωπικό του υπολογιστή, λίγες από τις γνωστές εταιρίες δίνουν την δυνατότητα ανάλυσης της επισκεπτόμενης ιστοσελίδας. Οι περισσότερες προσφέρουν ασφάλεια κατά του phishing (καθώς και ασφαλείς συναλλαγές), μέσω κάποιου δικού τους browser που κρυπτογραφεί πολύ καλά την πληροφορία που ανταλλάσσεται. Προστατεύουν δηλαδή από "ψεύτικες" ιστοσελίδες που σκοπό έχουν να προσελκύσουν ανυποψίαστα θύματα και να τους αποσπάσουν προσωπικές πληροφορίες όπως αριθμούς πιστωτικών ή χρεωστικών καρτών, κωδικούς πρόσβασης, αριθμούς τραπεζικών λογαριασμών, τηλέφωνα, διευθύνσεις κ.α.

Βέβαια υπάρχουν και εφαρμογές που εκτελούν τη διαδικασία που θέλουμε, όπως το AVG Antivirus και το AVG Internet Security, τα οποία όμως ζητούν ετήσια συνδρομή και υποχρεώνουν το χρήστη στο να πληρώνει με σκοπό να προστατεύεται από τις κακόβουλες ιστοσελίδες κατά την περιήγησή του στο διαδίκτυο. Όσον αφορά δωρεάν εφαρμογές ανάλυσης ιστοσελίδων, η απάντηση είναι, ναι υπάρχουν. Πρόκειται για Web εργαλεία στα οποία πρέπει κάποιος, χειροκίνητα, να δώσει κάποια είσοδο. Να κάνει δηλαδή αντιγραφή - επικόλληση κάποιο σύνδεσμο, κώδικα ή να ανεβάσει κάποιο αρχείο έτσι ώστε η εφαρμογή να το αναλύσει, να το ελέγξει και να του επιστρέψει τα αποτελέσματα υπό την μορφή κάποιου report.

Τα εργαλεία αυτά είναι τα:

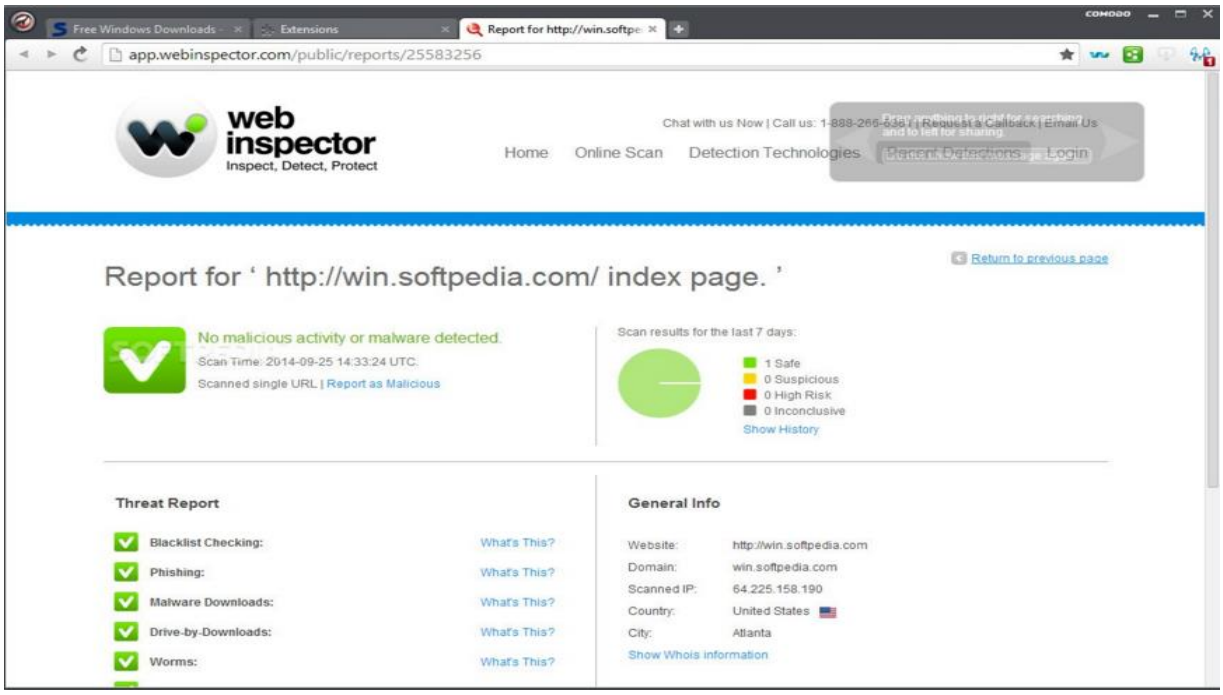
- Web Inspector (Comodo)
- JsDetox
- WepAWet
- JsUnpack
- Malzilla

3.1 Web-Inspector

Το Web-inspector είναι μια cloud-based υπηρεσία, η οποία προστατεύει την ιστοσελίδα μιας εταιρείας από καθημερινές επιθέσεις, ελέγχοντας τα τρωτά της σημεία. Ουσιαστικά πρόκειται για μια εφαρμογή που ενσωματώνει έλεγχο των σελίδων που δέχονται πληρωμές για την συμμόρφωσή τους με τα διεθνή πρότυπα. Ακόμη, περιλαμβάνεται η δυνατότητα σάρωσης του Server, κατά την οποία μια εταιρεία μπορεί να δώσει στην Comodo άμεση πρόσβαση στα αρχεία και τον πηγαίο κώδικα μέσω FTP ή SSH πρωτοκόλλου. Με αυτήν την πρόσβαση ελέγχονται :

- Λανθασμένα δικαιώματα πρόσβασης σε αρχεία ή φακέλους
- .htaccess hacks (τα .htaccess είναι πρόσθετα configuration αρχεία για web servers)
- κακόβουλα php αρχεία που επιτρέπουν shell-shock attacks
- Phishing pages
- Webpage Defacements

Το Web – Inspector είναι μία λύση που δεν διατίθεται δωρεάν αλλά δεν είναι κατάλληλη ούτε για οικιακή χρήση. Η Comodo όμως έχει κοινοποιήσει μια Web εφαρμογή, την οποία μπορεί να χρησιμοποιήσει ο οποιοσδήποτε δεν είναι σίγουρος για κάποια ιστοσελίδα και θέλει να την ελέγξει. Πολύ απλά, πλοηγείται στην εφαρμογή και δίνει ως είσοδο το url της σελίδας που θέλει να ελέγξει. Η Comodo εκτελεί τους ελέγχους που πρέπει κ επιστρέφει 1 ολοκληρωμένο malware report. Καθ' αυτόν τον τρόπο μπορεί ο οποιοσδήποτε χρήστης του διαδικτύου να ελέγξει μια σελίδα που θέλει να επισκεφτεί.



Εικόνα 1 - Ο ιστότοπος για το Web Inspector της Comodo

3.2 JsDetox

Το JsDetox είναι ένα εργαλείο ανάλυσης malware JavaScript. Χρησιμοποιεί τεχνικές όπως deobfuscation και στατική ανάλυση κώδικα, καθώς επίσης και εκτέλεση του κώδικα σε ασφαλές περιβάλλον, εξομοιώνοντας το DOM της HTML. Σαν user-interface χρησιμοποιεί κάποιον browser όμως όλη η ανάλυση και εκτέλεση του κώδικα γίνεται στο backend.

JsDetox Static Analysis

Το JsDetox δεν επαναδιαμορφώνει απλά τον κώδικα. Τον αναλύει και είναι ικανό να προϋπολογίσει τα στατικά του μέρη.

Αρχικός κώδικας
var x = 10 * 3 + 100 -70 /10;

analysis report
var x = 123;

Εξομοίωση HTML DOM

Τα JavaScript malware εκτός της τεχνικής του obfuscation, που χρησιμοποιούν για να κρύψουν τον κώδικά τους, χρησιμοποιούν και αντικείμενα και μεθόδους που μπορεί κανείς να βρει μόνο μέσα σε κάποιον browser, όπως το αντικείμενο "document". Το JsDetox, όπως αναφέρθηκε

προηγουμένως, εξομοιώνει μέρη και λειτουργίες ενός browser και καταφέρνει να αναλύσει πιο "δυναμικά" μέρη ενός κακόβουλου κώδικα.

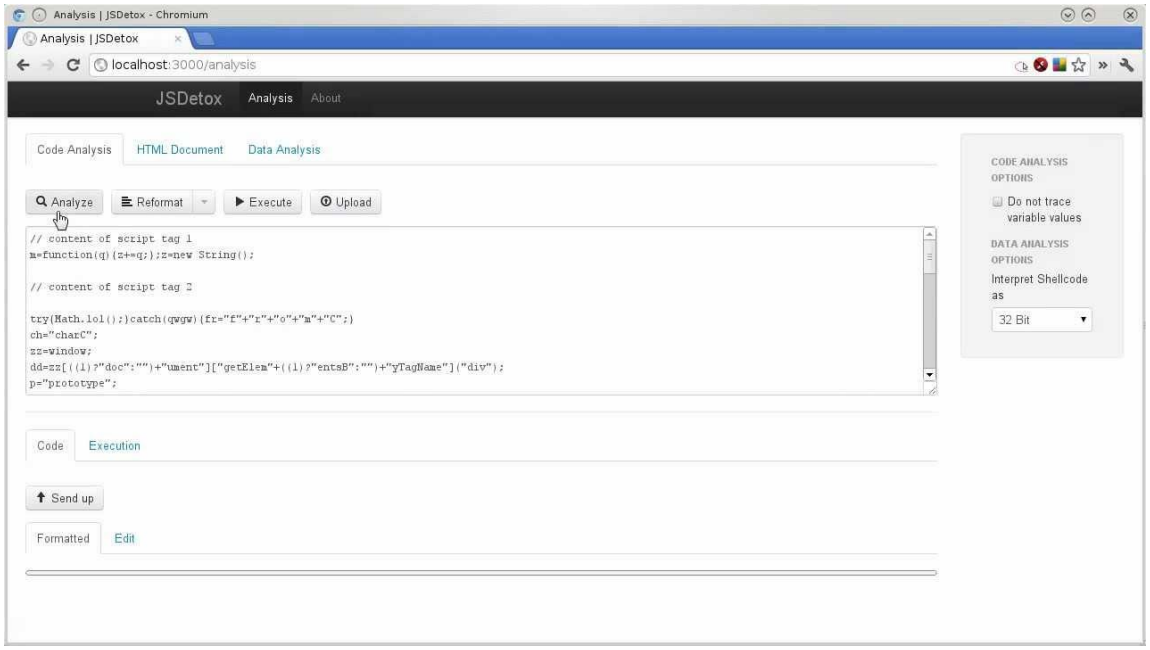
```
document.write ('<div id="A123">212</div>');
var tZkl = parseInt(document.getElementById("A123").innerHTML;
if (tZkl == 212 ) {
....
...
}
```

Data Analysis

Το JsDetox μπορεί ακόμα να χρησιμοποιηθεί για την ανάλυση shellcode που κρύβεται σε κάποιο κώδικα malware JavaScript. Οι περισσότεροι κώδικες malware κρύβονται σε μοναδικές ακολουθίες όπως:

```
%u4141%u4141%u8366%ufce4%uebfc .....
```

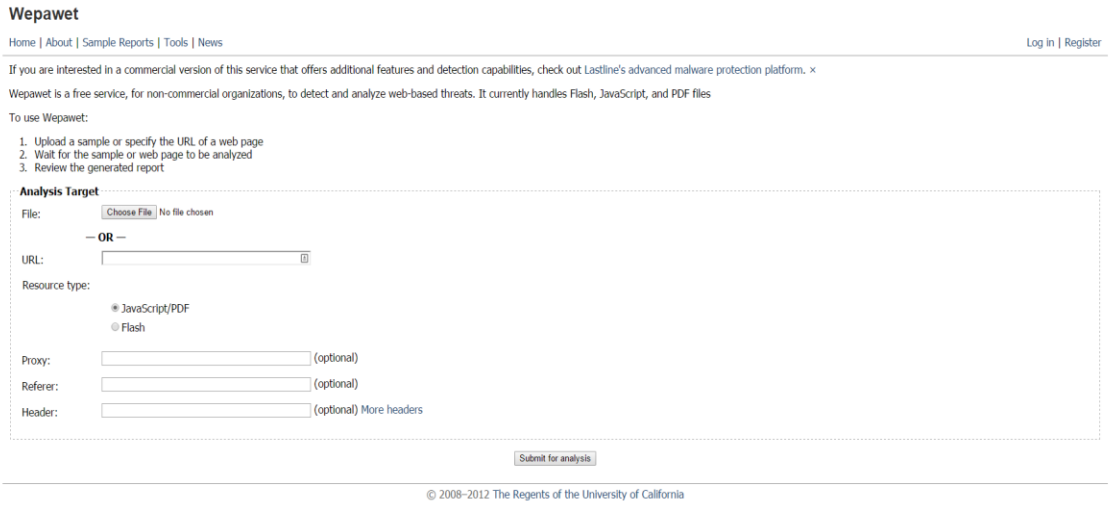
Με τη δυνατότητα αυτή το JsDetox μπορεί να χρησιμοποιηθεί για να αναλύσει τέτοιες συμβολοσειρές και να εξάγει από αυτές το shellcode. Το εξαχθέν shellcode, μπορεί κανείς να το "διαβάσει" σε HexDump ή Disassembled Code. Τα περισσότερα shellcodes περιέχουν δεδομένα κρυπτογραφημένα με επανάληψη διαδικασίας XOR και συνήθως περιέχουν το url από το οποία θα κατεβάσουν το πραγματικό malware.



Εικόνα 2 - Το πρόγραμμα JsDetox

3.3 WepAWet

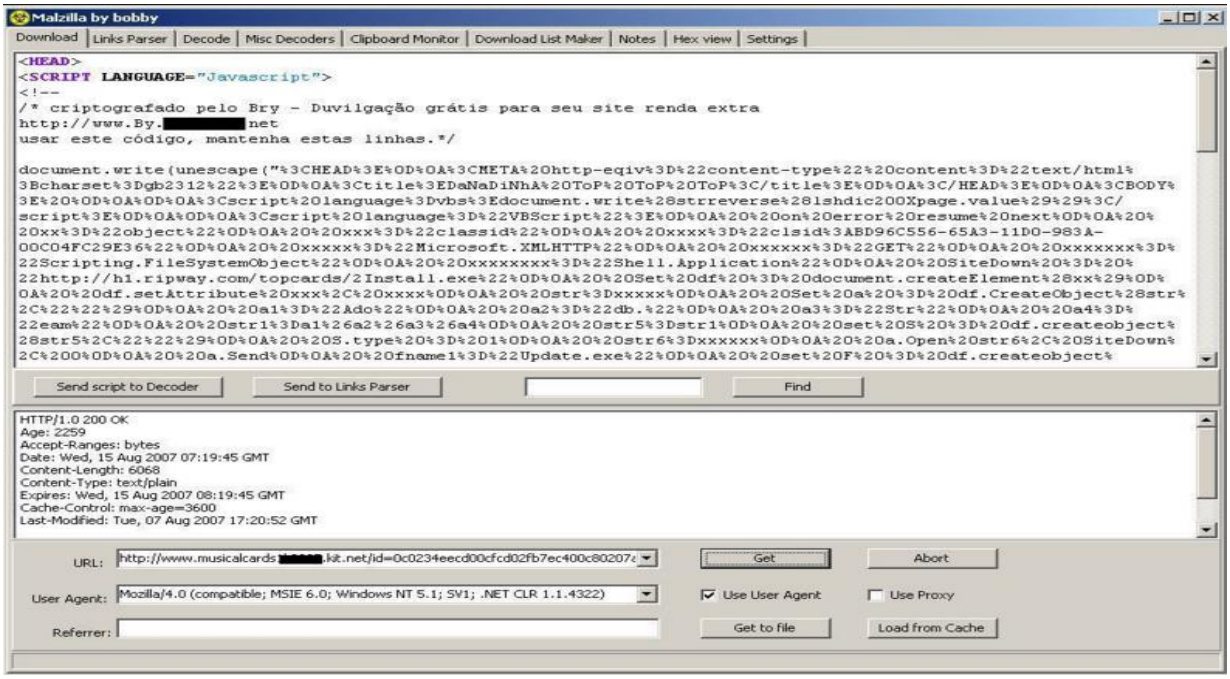
Το WepAWet είναι μια ιστοσελίδα η οποία κατασκευάστηκε και συντηρείται από το τμήμα επιστήμης υπολογιστών του University of California, Santa Barbara. Δεδομένης μιας ιστοσελίδας, το wepawet, αναλύει κώδικα Javascript ή Flash για κακόβουλο περιεχόμενο. Όπως και το jsDetox, το wepawet δίνει σε κάποιον την δυνατότητα να "ανεβάσει" κάποιο αρχείο με κώδικα ή να δώσει κάποιο url και σαν αποτέλεσμα να λάβει μια λεπτομερή αναφορά με την ανάλυση της ιστοσελίδας. Το εν λόγω report περιέχει σημαντικές πληροφορίες για το ποια σημεία του ανεβασμένου κώδικα/αρχείου είναι κακόβουλα καθώς επίσης μπορεί και να επιστρέψει πληροφορίες για τις ευπάθειες που εκμεταλλεύονται τα σημεία αυτά. Η στατική ανάλυση που εκτελεί το wepawet ψάχνει σε ύποπτα κομμάτια κώδικα ή κλήσεις συναρτήσεων για κακόβουλο περιεχόμενο. Επιπλέον κάθε υποβληθέν αρχείο, υποβάλλεται επίσης στο VirusTotal (link) και τα αποτελέσματα εμπεριέχονται στην περίληψη της ανάλυσης.



Εικόνα 3 - Ο ιστότοπος του Wepawet

3.4 Malzilla

Το malzilla είναι ένα εργαλείο για Windows που αναζητά και αυτό κακόβουλο περιεχόμενο. Το malzilla δέχεται ως είσοδο ένα URL ή τον κώδικα προς έλεγχο αυτούσιο. Στην περίπτωση του URL, το malzilla επιτρέπει να θέσει κάποιος διαφορετικούς HTTP Headers όπως πχ, User-Agent και Referrer. Ακόμη, έχει την δυνατότητα εξαγωγής του Javascript κώδικα που βρίσκεται μέσα στο DOM της σελίδας (inline) και όχι σε κάποιο ξεχωριστό αρχείο *.js . Επίσης περιέχει μια μεγάλη γκάμα από decoders και deobfuscators καθώς τα περισσότερα websites χρησιμοποιούν διαφορετικές τεχνικές obfuscation για να κρύψουν τον πραγματικό ρόλο του κώδικά τους ή το url από το οποίο θα λάβουν επιπλέον δεδομένα.



Εικόνα 4 - Το περιβάλλον χρήσης του Malzilla

3.5 Js-Unpack-n

Το js-unpack διατίθεται σε 2 εκδόσεις, είτε ως web εφαρμογή, είτε ως command-line εργαλείο για λειτουργικά συστήματα linux. Δίνει την δυνατότητα στον χρήστη να προσδιορίσει 1 ή περισσότερα URLs τα οποία θα επεξεργαστεί με σκοπό την διερεύνηση του ενσωματωμένου τους κώδικα. Στόχος του js-unpack είναι η αναγνώριση των λειτουργιών εισόδου/εξόδου του κώδικα αλλά και επιθέσεις που προσπαθούν να εκμεταλλευτούν ευπάθειες του ίδιου του browser ή και κάποιου plugin του. Πιο λεπτομερειακά, το js-unpack περιέχει ένα sandbox το οποίο εξομοιώνει την λειτουργία ενός browser για να αναλύσει την συμπεριφορά κάποιου αντικειμένου (Javascript κώδικα, PDF, OCAP, HTML, SWF κ.α.). Αποκρυπτογραφεί τον κρυπτογραφημένο Javascript κώδικα και τον εκτελεί σε ασφαλές περιβάλλον, όπως ακριβώς θα έκανε ένας οποιοσδήποτε κοινός browser.

4. Μέσα περιήγησης στο διαδίκτυο (browsers)

Οι περιηγητές ιστού αποτελούν τα δημοφιλέστερα προγράμματα για είσοδο και περιήγηση του χρήστη στον κόσμο του διαδικτύου. Παρακάτω θα αναφέρουμε αρκετούς περιηγητές και θα αναλύσουμε τους σημαντικότερους από αυτούς.

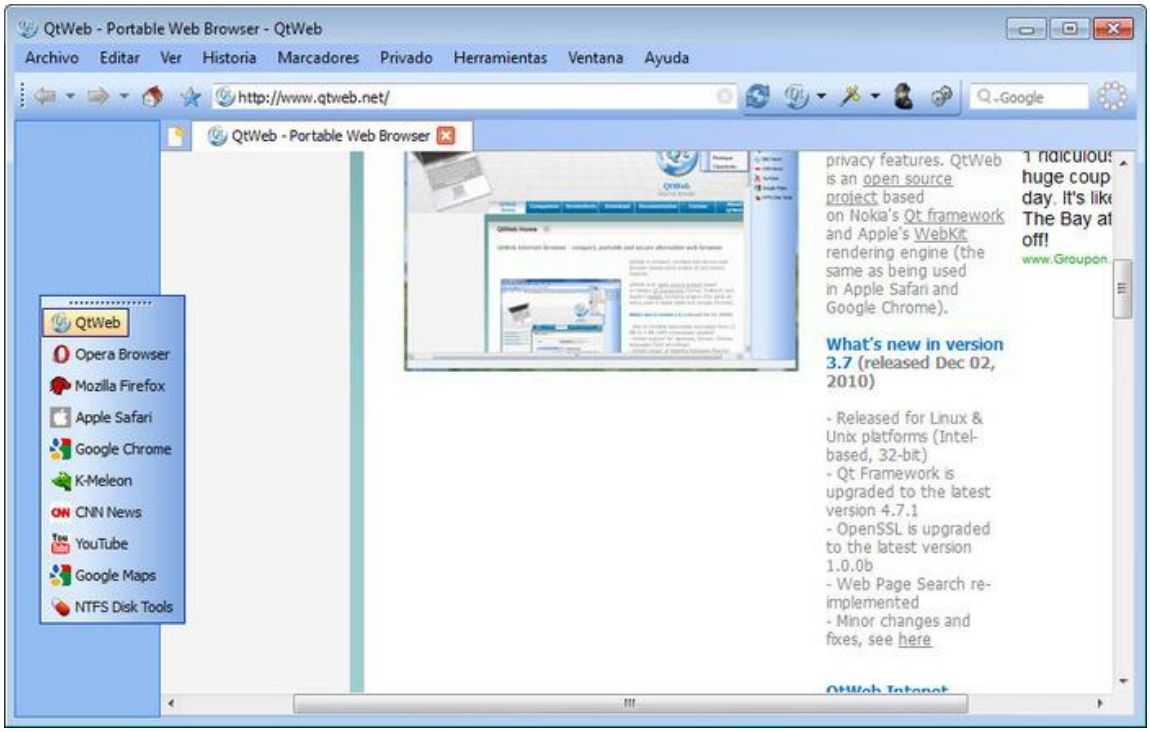
4.1 RockMelt

Ο RockMelt είναι ένας διαδικτυακός περιηγητής που σχεδιάστηκε και υλοποιήθηκε από τους Tim Howes και Eric Vishria, πρώην μηχανικούς ανάπτυξης του πολύ δημοφιλούς browser, Netscape Navigator και βασίστηκε στη πλατφόρμα του Google Chromium. Ο RockMelt δίνει ιδιαίτερη βαρύτητα στη χρήση των κοινωνικών δικτύων Facebook και Twitter. Το περιβάλλον χρήσης του, είναι διαμορφωμένο με τέτοιο τρόπο, ώστε να έχει ο χρήστης πρόσβαση σε πολλά από τα βασικά στοιχεία του κοινωνικού δικτύου Facebook αλλά και του Twitter ανεξαρτήτως το σε ποια ιστοσελίδα βρίσκεται. Επίσης προσφέρει τη δυνατότητα αναζήτησης στο Google χωρίς να χρειάζεται να επισκεφτεί ο χρήστης τη σελίδα της μηχανής αναζήτησης. Υποστηρίζει τα λογισμικά Windows και Mac και λειτουργεί και σε συσκευές iOS. Το 2013 η εταιρεία Yahoo εξαγόρασε το RockMelt και το απέσυρε από την αγορά με σκοπό να χρησιμοποιήσει τις εξελιγμένες λειτουργίες του σε άλλα μελλοντικά της προϊόντα.

Εικόνα 5 - Ο περιηγητής RockMelt

4.2 QTweb

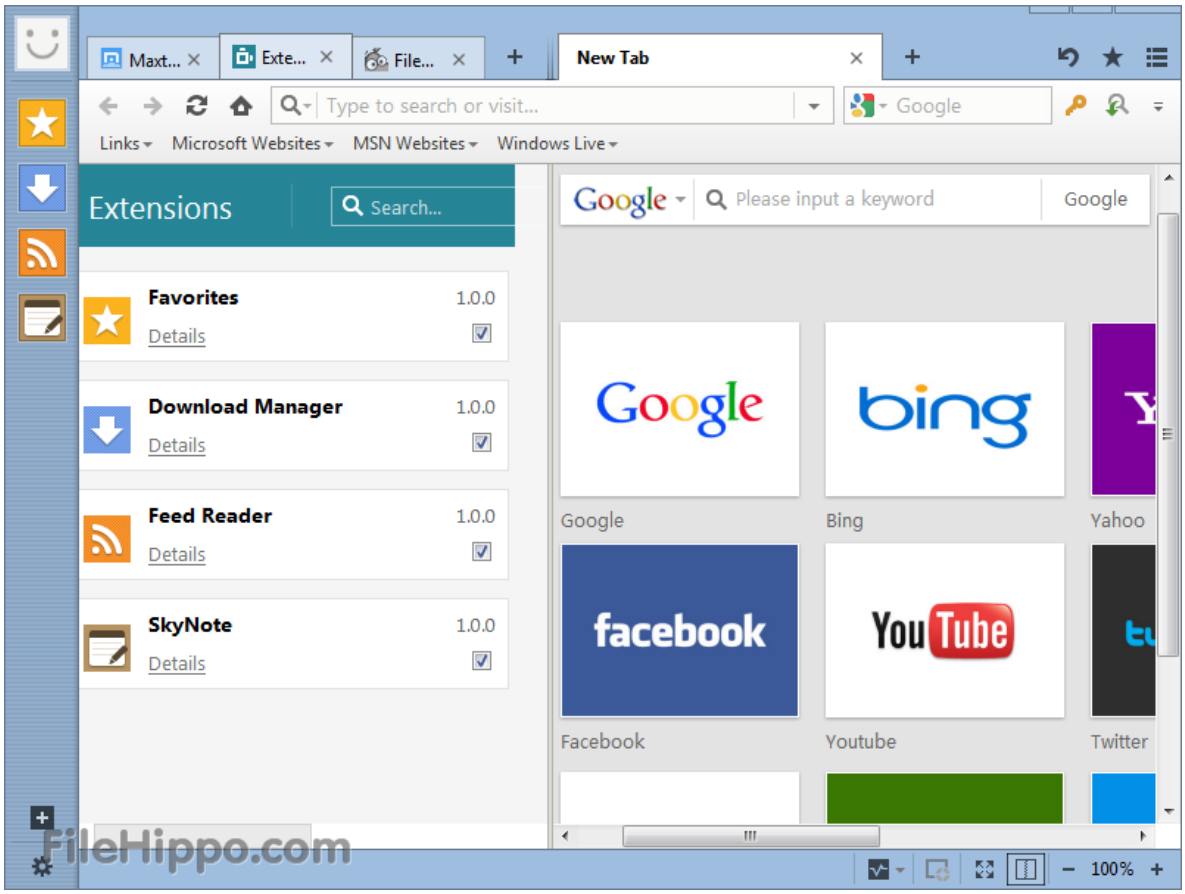
Το QTweb είναι ένα πρόγραμμα περιήγησης του διαδικτύου που είναι εύχρηστο, ασφαλές και δεν απαιτεί εγκατάσταση. Ουσιαστικά μπορείτε να το έχετε αποθηκευμένο ακόμα και σε ένα USB stick και να το τρέχετε από εκεί σε οποιονδήποτε υπολογιστή. Καλές επιδόσεις ταχύτητας και μικρές.



Εικόνα 6 - Ο περιηγητής QTweb

4.3 Maxthon

Ο Maxthon είναι ένας cloud based περιηγητής που βασίζεται στη τεχνολογία του Internet Explorer. Προσφέρει tabbed browsing καθώς και συμβατότητα με τις περισσότερες νέες τεχνολογίες. Περιέχει καλά εργαλεία και συστήματα προστασίας από κακόβουλες σελίδες/προγράμματα. Ο Maxthon έρχεται εξοπλισμένος με αρκετές δυνατότητες που δύσκολα βρίσκονται σε άλλους περιηγητές όπως το Resource Sniffer που μπορεί και «κατεβάζει» τα πολυμέσα που έχει μια ιστοσελίδα. Επιπλέον διαθέτει το Magic Fill, έναν διαχειριστή κωδικών για πολλαπλούς λογαριασμούς, το Snap το οποίο είναι ένα εργαλείο αποτύπωσης οθόνης. Δίνει ακόμη την δυνατότητα προσθήκης συντόμευσης οποιουδήποτε προγράμματος στην μπάρα εργαλείων.

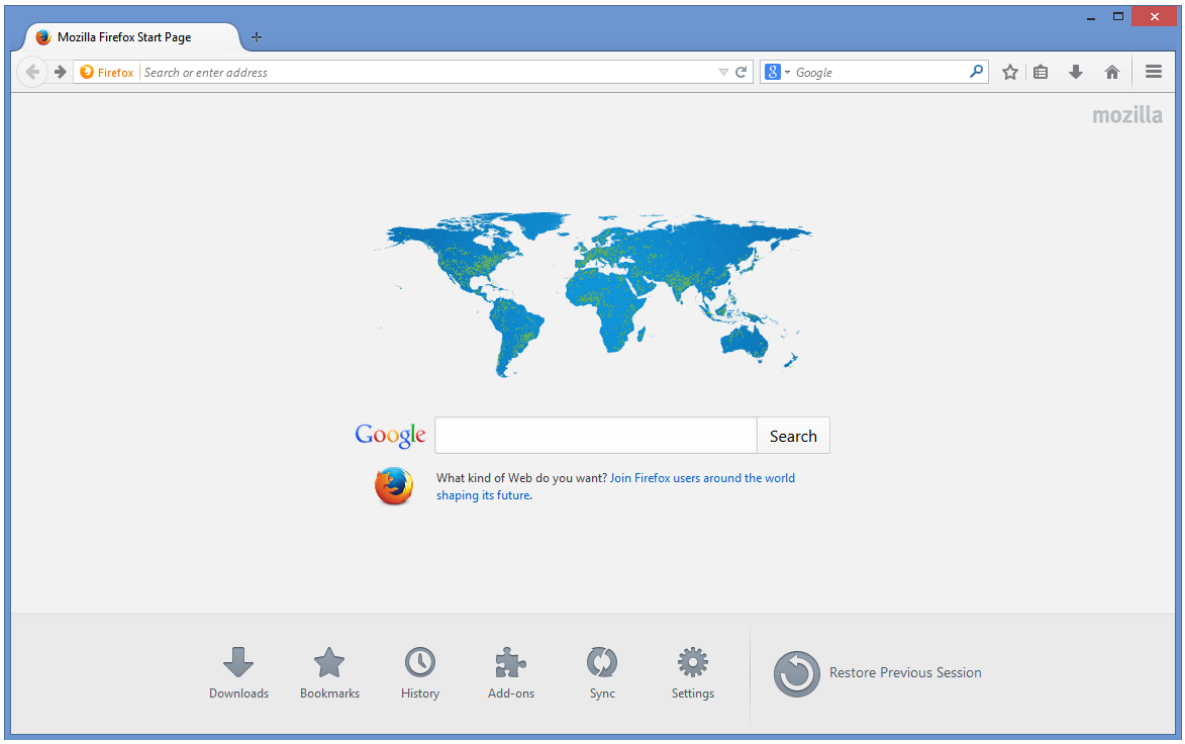


Εικόνα 7 – Ο cloud-based περιηγητής Maxthon

4.4 Firefox

Ο Mozilla Firefox είναι ένας δωρεάν και ανοιχτού κώδικα φυλλομετρητής του παγκόσμιου ιστού. Προήλθε από την σουίτα εφαρμογών της Mozilla και η ανάπτυξή του εξακολουθεί να γίνεται, σε πολύ μεγάλο ποσοστό από την Mozilla Corporation ενώ ταυτόχρονα συνεισφέρουν και μεμονωμένοι προγραμματιστές. Ο Firefox είναι ο τρίτος σε χρήση, στον κόσμο, περιηγητής διαδικτύου πίσω από τον Google Chrome και τον Internet Explorer. Στις λειτουργίες του περιλαμβάνονται ο συγχρονισμός των δεδομένων του χρήστη στις διάφορες διαφορετικές του συσκευές και προστασία παρακολούθησης σε κατάσταση ασφαλούς περιήγησης. Πλέον πολλοί ιστότοποι έχουν γίνει αδύνατο για κάποιο χρήστη να τους διαβάσει, έτσι ο Firefox δημιούργησε το «Reading Mode», κατά το οποίο ο Firefox παρουσιάζει στον χρήστη το πραγματικό περιεχόμενο της ιστοσελίδας. Μια επιπλέον πολύ καλή προσθήκη που έχει κάνει ο Firefox είναι το Firefox Hello. Το Firefox Hello είναι μια ενσωματωμένη εφαρμογή βιντεοκλήσεων.

Ο Firefox είναι ίσως ο καλύτερος περιηγητής διαδικτύου αυτή την στιγμή. Είναι λιτός, λειτουργικός και προσαρμόσιμος, ενώ προσφέρει τον καλύτερο συνδυασμό απόδοσης, χαρακτηριστικών, υποστήριξης για νέα Web πρότυπα, χαμηλή χρήση μνήμης και προστασία ιδιωτικότητας.



Εικόνα 8 - Ο πολύ γνωστός περιηγητής Mozilla

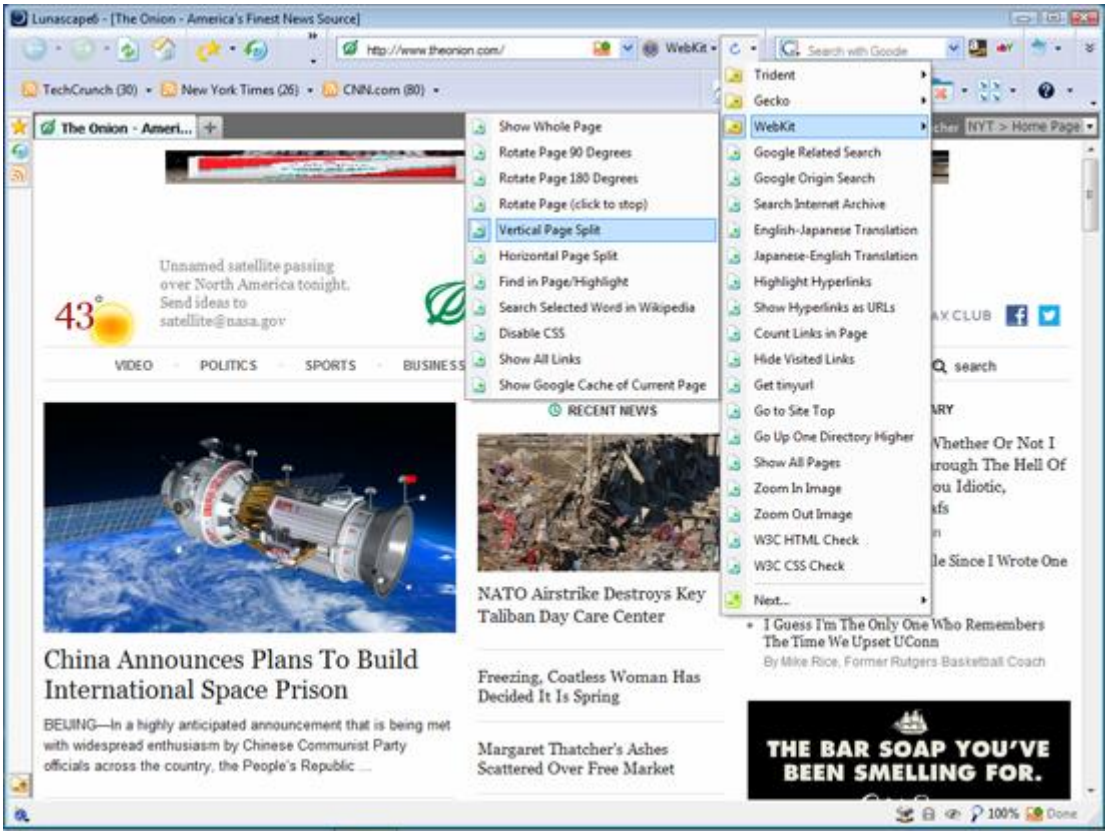
4.5 Safari

Μέχρι το 1997 οι υπολογιστές της Apple είχαν προεγκατεστημένο τον Netscape Navigator και τον Cyberdog. Στη συνέχεια μετά από μία συνεργασία για 5 χρόνια της Microsoft με την Apple, οι υπολογιστές Mac είχαν προεγκατεστημένο τον Internet Explorer για Mac. Τότε, σε εκείνο το διάστημα η Apple αποφάσισε να προβεί στη δημιουργία του δικού της browser. Αυτός είναι ο Safari. Τον Ιανουάριο του 2003 παρουσιάστηκε για πρώτη φορά ο περιηγητής Safari ο οποίος βασίστηκε στο Webkit που χρησιμοποιούσε μέχρι τότε η ίδια η εταιρεία για εσωτερική χρήση. Σε κάθε νέο υπολογιστή Apple, ήταν προεγκατεστημένος και δεν άργησε να αποκτήσει μεγάλο φανατικό κοινό. Έκτοτε ο Safari αποτελεί σημαντικό μέρος του MacOS και εξελίσσεται σταθερα στο χρόνο.

Εικόνα 9 - Ο περιηγητής της Apple, Safari

4.6 Lunascape

Πρόγραμμα περιήγησης διαδικτυακών σελίδων με μια ιδιαιτερότητα. Επιτρέπει την εμφάνιση των σελίδων όπως αυτές φαίνονται σε τρεις διαφορετικούς (τους πιο δημοφιλείς για την ακρίβεια) άλλους περιηγητές. Ο μόνος φυλλομετρητής που δεν στηρίζεται ένα και μόνο πυρήνα, όπως ο Internet Explorer, ο Google Chrome και ο Mozilla Firefox αλλά και στους τρεις ταυτόχρονα. Εξυπηρετεί webmasters και σχεδιαστές που θέλουν γρήγορα να δουν πως φαίνεται η δουλειά τους στον Internet Explorer, τον Firefox και τον Google Chrome. Ο χρήστης έχει την δυνατότητα να επιλέξει να θέση κάποιον από τους τρεις πυρήνες ως προεπιλεγμένο, με τον οποίο και θα ξεκινά όταν ανοίγει ο φυλλομετρητής ή όταν ανοίγει ο χρήστης μια νέα καρτέλα. Μέσω εντός κουμπιού στην γραμμή εργαλείων, ο χρήστης μπορεί να εναλλάσσει τους πυρήνες για την καρτέλα την οποία βρίσκεται έτσι ώστε να ανακαλύψει εάν η ιστοσελίδα φαίνεται σωστά και στους τρεις διαφορετικούς φυλλομετρητές ή αν κάποια δυνατότητα της σελίδας δεν τρέχει σωστά σε κάποιον. Δυστυχώς αυτή η δυνατότητα δεν λειτουργεί τέλεια καθότι οι τρεις πυρήνες στους οποίους βασίζεται ο Lunascape, δεν είναι στην τελευταία τους έκδοση. Αυτό το γεγονός καθιστά τον Lunascape όχι και τόσο αξιόπιστο για την λειτουργία για την οποία και δημιουργήθηκε.

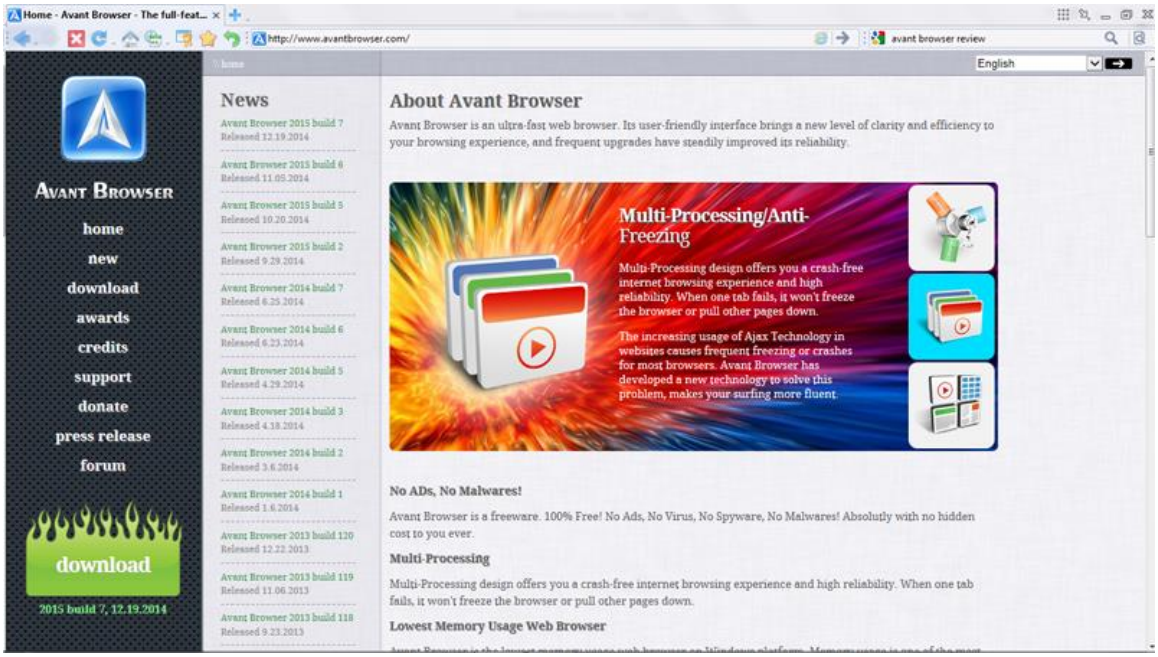


Εικόνα 10 - Ο περιηγητής Lunascape

4.7 Avant

Ο Avant είναι ένας φυλλομετρητής που όπως και ο Lunascape προσφέρει την δυνατότητα της εναλλαγής πυρήνων για την προβολή κάποιας ιστοσελίδας. Προσφέρει δυνατότητες όπως αυτόματες ενημερώσεις, διαχείριση κωδικών και μια ενσωματωμένη μηχανή αναζήτησης του Internet Explorer. Χρησιμοποιεί επίσης ένα σχέδιο πολύ-διεργασιών, που σημαίνει ότι όλες οι ανοιχτές καρτέλες λειτουργούν ανεξάρτητα η μια από την άλλη και στην περίπτωση που κάποια σταματήσει να λειτουργεί δεν θα επηρεαστεί ολόκληρος ο περιηγητής. Δυστυχώς ο Avant δεν έχει την δυνατότητα του γονικού ελέγχου (parental control) όπως και δεν είναι και η καλύτερη λύση όσον αφορά την ασφάλεια στο διαδίκτυο. Έχει τα ίδια τρωτά σημεία με τον Internet Explorer, αφού ουσιαστικά πρόκειται για μια επέκτασή του. Ο Avant είναι αρκετά γρήγορος περιηγητής αλλά πολλές φορές δυσκολεύεται να ανταπεξέλθει στις απαιτήσεις των σημερινών διαδικτυακών εφαρμογών.

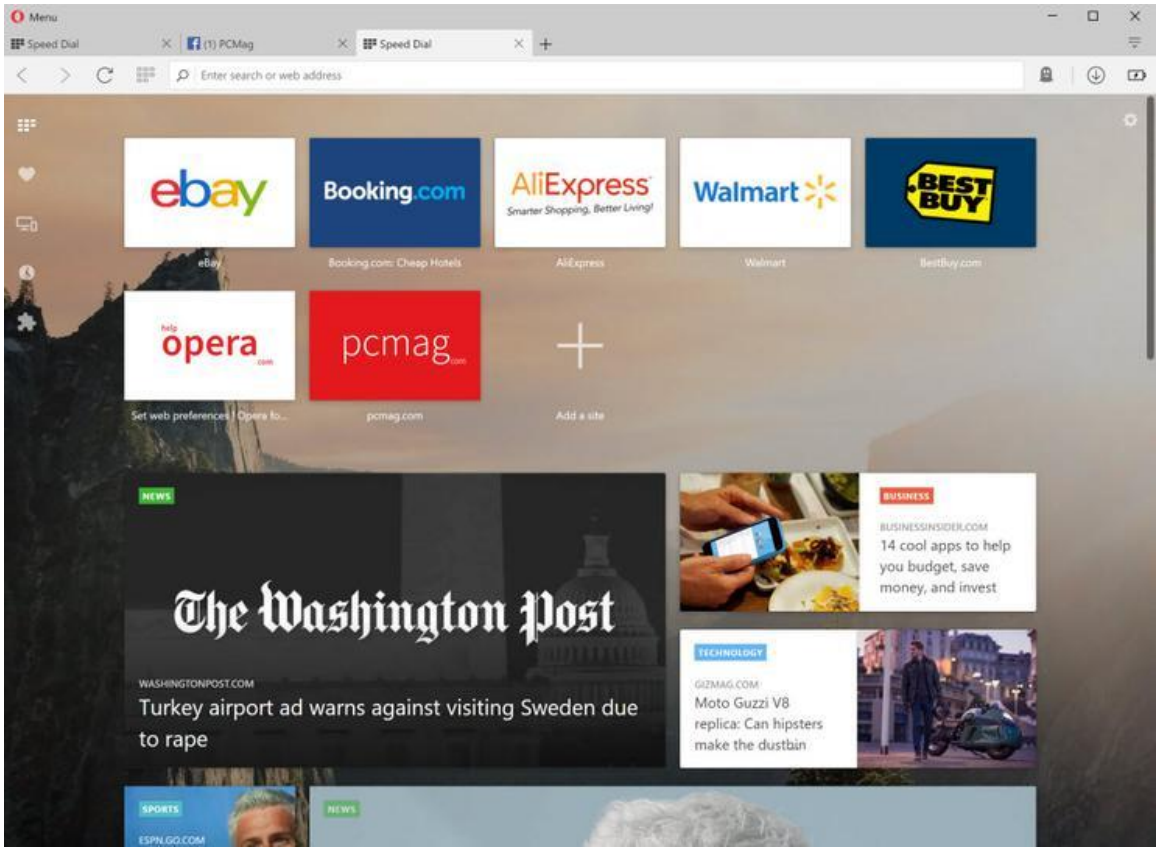
Ο Avant είναι ένας αξιοπρεπής φυλλομετρητής με μερικά ωραία ενσωματωμένα χαρακτηριστικά και επιλογές προσαρμογής στο γραφικό περιβάλλον. Ωστόσο, πάσχει σε θέματα ευελιξίας και ασφάλειας τα οποία συμβάλλουν αρκετά στο να μην είναι ανταγωνιστικός σε σχέση με τους πιο γνωστούς περιηγητές (Internet Explorer, Mozilla Firefox, Google Chrome κ.α.). Είναι όμως μια πολύ καλή επιλογή για κάποιον που θέλει να χρησιμοποιήσει έναν φυλλομετρητή βασισμένο στον Internet Explorer αλλά έχει δεν του αρέσουν οι τρέχουσες ρυθμίσεις που αυτός προσφέρει.



Εικόνα 11 - Ο περιηγητής Avant

4.8 Opera

Πρόκειται για μια από τις καλύτερες εναλλακτικές λύσης στον χώρο. Είναι δωρεάν και προσφέρει πια πληθώρα από εργαλεία που θα φανούν χρήσιμα στους περισσότερους χρήστες. Δεν έχει προβλήματα συμβατότητας, αφού χρησιμοποιεί τον ίδιο πυρήνα με τον Google Chrome που είναι συμβατός σχεδόν με οτιδήποτε κυκλοφορεί στο διαδίκτυο. Προσφέρει ένα αρκετά ωραίο και προσιτό γραφικό περιβάλλον καθώς και δυνατότητες εύκολης χρήσης όπως το Speed dial, το Video Pop out και συγχρονισμό μεταξύ πολλαπλών συσκευών. Παρέχει ακόμη αρκετά καινοτόμα και χρήσιμα χαρακτηριστικά όπως η επιλογή Turbo. Στην επιλογή αυτή, ο περιηγητής χρησιμοποιεί την ίδια διαδικασία προσωρινής μνήμης και συμπίεσης των σελίδων με τον Opera mini, που είναι ο πιο διαδεδομένος φυλλομετρητής για κινητές συσκευές (smartphones, tablets κτλ.) και είναι ιδανικό για όσους δεν έχουν πολύ γρήγορη σύνδεση στο διαδίκτυο. Από πλευράς ασφάλειας και ιδιωτικότητας, ο Opera, έχει κάποια πολύ καλά χαρακτηριστικά όπως το sandbox του Google Chrome για να «τρέχει» κάθε διαδικτυακή εφαρμογή σε δικό της απομονωμένο περιβάλλον, το επαναστατικό built-in Ad Blocker που αφαιρεί τις διάφορες διαφημίσεις από τις σελίδες που επισκέπτεται ο χρήστης βελτιώνοντας τους χρόνους απόκρισης των ιστοσελίδων αλλά και προστατεύοντας τον χρήστη από διάφορους κινδύνους που κρύβονται πίσω από αυτές. Ο Opera προσφέρει, επίσης, ενσωματωμένο το λεγόμενο Operatic VPN. Το Operatic VPN, είναι μια δωρεάν VPN υπηρεσία που προσφέρει ο Opera. Μέσω του VPN, ο χρήστης μπορεί να προστατευτεί από επιτιθέμενους που έχουν σκοπό την κλοπή των δεδομένων ή την προώθηση του χρήστη σε ψευδείς κακόβουλους ιστότοπους.

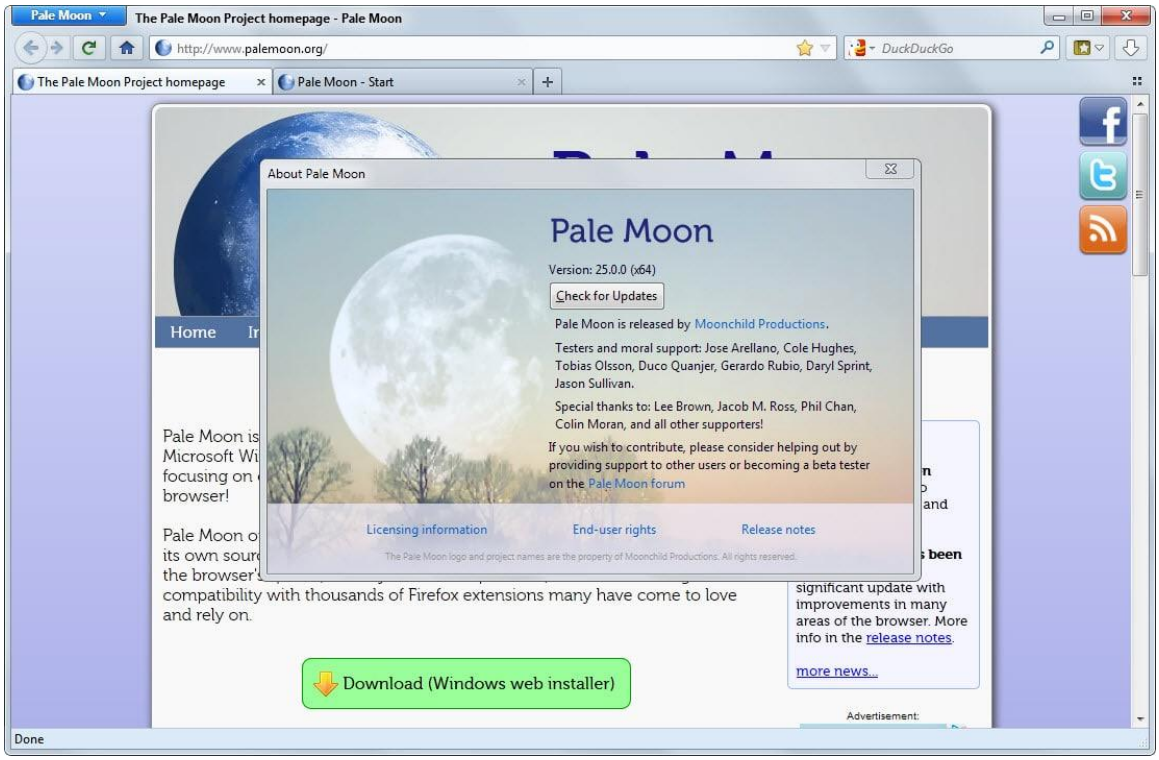


Εικόνα 12 - Ο διάσημος περιηγητής Opera

4.9 Pale Moon

Ο Pale Moon είναι ένας διαδικτυακός Περιηγητής βασισμένος στον κώδικα του Mozilla Firefox αλλά ειδικά τροποποιημένος για να τρέχει καλύτερα σε λειτουργικά συστήματα Windows. Είναι παρακλάδι ενός παλιού κώδικα του Firefox ESR αλλά ιδιαιτέρως τροποποιημένο. Χαρακτηριστική είναι η άρνηση των δημιουργών του να ενσωματώσουν τις αλλαγές του Firefox στον κώδικα, οι οποίες, στην πραγματικότητα, αφαιρούν λειτουργικότητα. Οι περισσότερες λειτουργίες του Firefox είναι εκεί καθώς και πλήρης συμβατότητα με τα Add-ons και τα Extensions του. Προσφέρει επίσης βελτιωμένη απόδοση για τους μοντέρνους κεντρικούς επεξεργαστές (CPU) καθώς και μικρότερες απαιτήσεις μνήμης (RAM).

Μια πρόσφατη συζήτηση στο κεντρικό forum του Pale Moon αναφέρει ότι η ομάδα ανάπτυξης σκέφτεται να δημιουργήσει έναν νέο φυλλομετρητή, παράλληλα με τον Pale Moon μέχρις ότου σταθεροποιηθεί και τελικά τον αντικαταστήσει. Ο νέος αυτός περιηγητής θα είναι βασισμένος σε νεότερη έκδοση του Firefox, χωρίς όμως να θυσιάζει τίποτα από το γραφικό περιβάλλον και τα χαρακτηριστικά που τον διαφοροποιούν από τον Firefox.



Εικόνα 13 - Ο περιηγητής Pale Moon

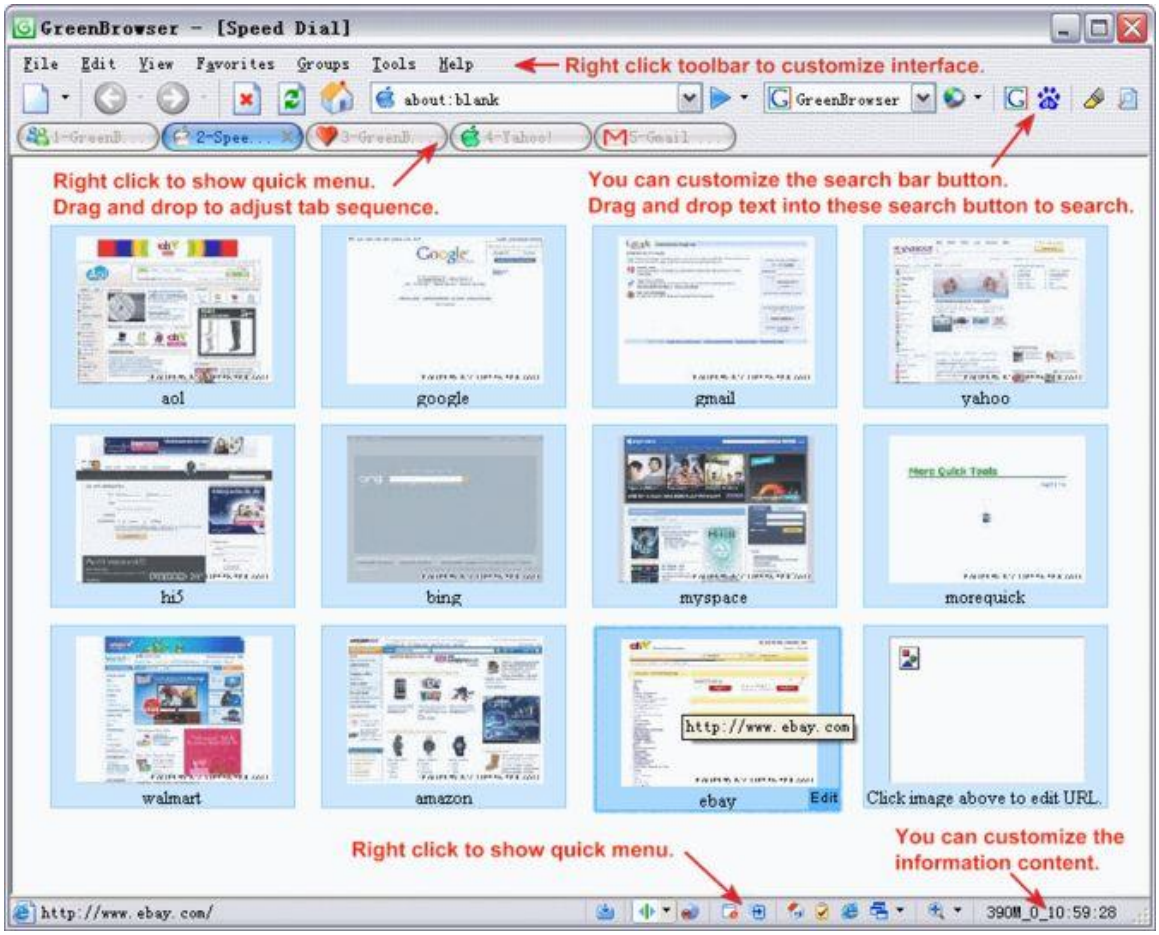
4.10 SongBird

Το SongBird είναι μια μίξη media player και internet browser που προσφέρει εξαιρετικές δυνατότητες και αναπαραγωγής ήχου και βίντεο από αρχεία που είναι αποθηκευμένα στον υπολογιστή σας, αλλά και τη δυνατότητα πλοήγησης του διεθνές διαδικτύου έχοντας ως κεντρικό άξονα τον ήχο και την εικόνα. Μέσω διαφόρων τεχνικών έχει προσφέρει τη δυνατότητα να συνδέσετε το πρόγραμμα με media blogs και άλλους σχετικούς κόμβους (όπως πχ το last.fm) δημιουργώντας έτσι τα δικά σας κανάλια εισροής οπτικοακουστικών ερεθισμάτων. Υποστηρίζει Windows και MacOS (και ανεπίσημα πλέον Linux) και συνδυάζει καλά το εύρος δυνατοτήτων με την ευκολία χρήσης.

4.11 GreenBrowser

Εφαρμογή περιήγησης του διεθνούς διαδικτύου που έχει τις βάσεις της στον κώδικά του Internet Explorer. Είναι αρκετά γρήγορος και ελαφρύς browser, ενώ έρχεται με πολλά έξτρα στοιχεία ενσωματωμένα. Για τον εξελιγμένο χρήστη είναι εύκολη διαδικασία να ξεκαθαρίσει τι θέλει να κρατήσει και τι όχι από όλα τα add-ons με τα οποία έρχεται προ-εγκατεστημένα. Από την άλλη ο αρχάριος χρήστης θα δυσκολευτεί να βγάλει άκρη, αν επιλέξει να αφαιρέσει κάποιο στοιχείο, κάτι το οποίο δεν είναι απαραίτητο βέβαια. Υποστηρίζεται και από μια συμπαθητική βιβλιοθήκη plugins τα οποία μπορεί ο χρήστης να εγκαταστήσει κατά βούληση, χωρίς να περιέχει όμως το εύρος άλλων browser (όπως Firefox, Chrome και Opera). Από άποψη χρήσης πόρων τα πάει αρκετά καλά (καλύτερα από αρκετούς άλλους browser), χωρίς να επιβαρύνει αισθητά το σύστημα ενώ η ταχύτητα φόρτωσης των σελίδων είναι επίσης πολύ καλή. Δυστυχώς το περιβάλλον χρήσης του

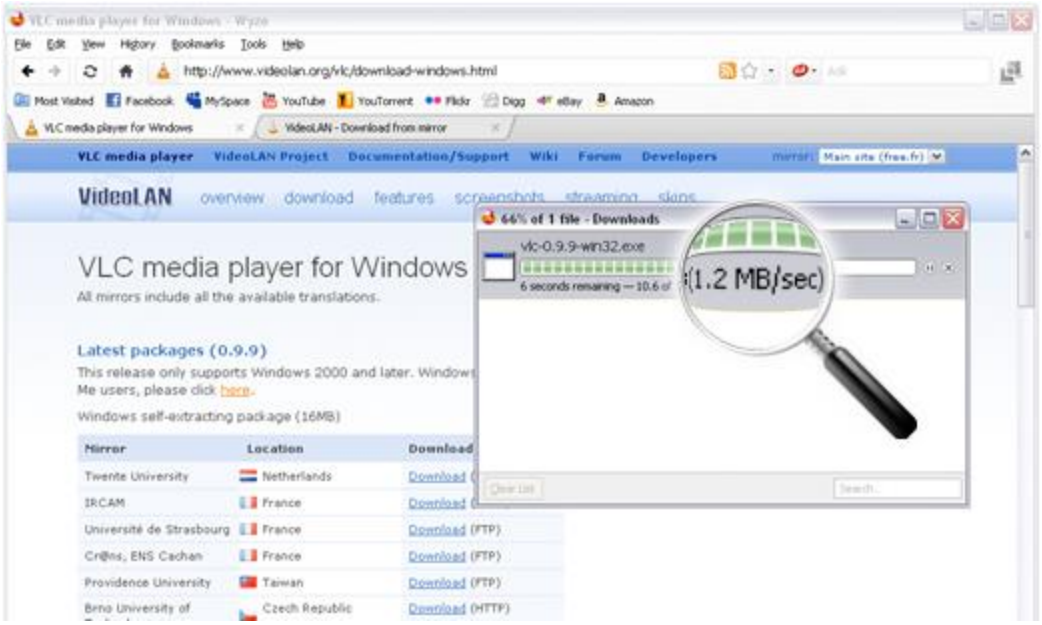
GreenBrowser δείχνει λίγο μονότονο και όχι ιδιαίτερα ελκυστικό παρά την δυνατότητα υποστήριξης θεμάτων.



Εικόνα 14 - Ο περιηγητής GreenBrowser

4.12 Wyzo

Ο Wyzo είναι ένας νέος browser που προσφέρει μια ενδιαφέρουσα και αξιόπιστη εναλλακτική λύση έναντι των καθιερωμένων. Καταρχήν, είναι βασισμένος στον κώδικά του Mozilla Firefox και υποστηρίζει όλα τα plugins του (καλό γιατί αλλιώς θα ήταν χιλιόμετρα πίσω από τους άλλους) ενώ παρέχει όλες τις κλασσικές λειτουργίες που έχει συνηθίσει στους φυλλομετρητές ο μέσος χρήστης. Προσφέρει επίσης ορισμένα επιπλέον εργαλεία, όπως ενσωματωμένο σύστημα διαχείρισης αρχείων torrent, ταχύτατη πλοήγηση σε ιστοσελίδες καθώς και βελτιστοποίηση και διαχείριση των μεταφορτώσεων σας (downloads). Οπτικά μοιάζει περισσότερο με το Chrome και λειτουργικά είναι πολύ κοντά στον Firefox. Για κάποιον χρήστη που τον ενδιαφέρουν οι επιπλέον λειτουργίες ή αν είναι χρήστης του Firefox, αλλά έχει προβλήματα με τις επιδόσεις του, ο Wyzo πρόκειται για μια πολύ καλή εναλλακτική.



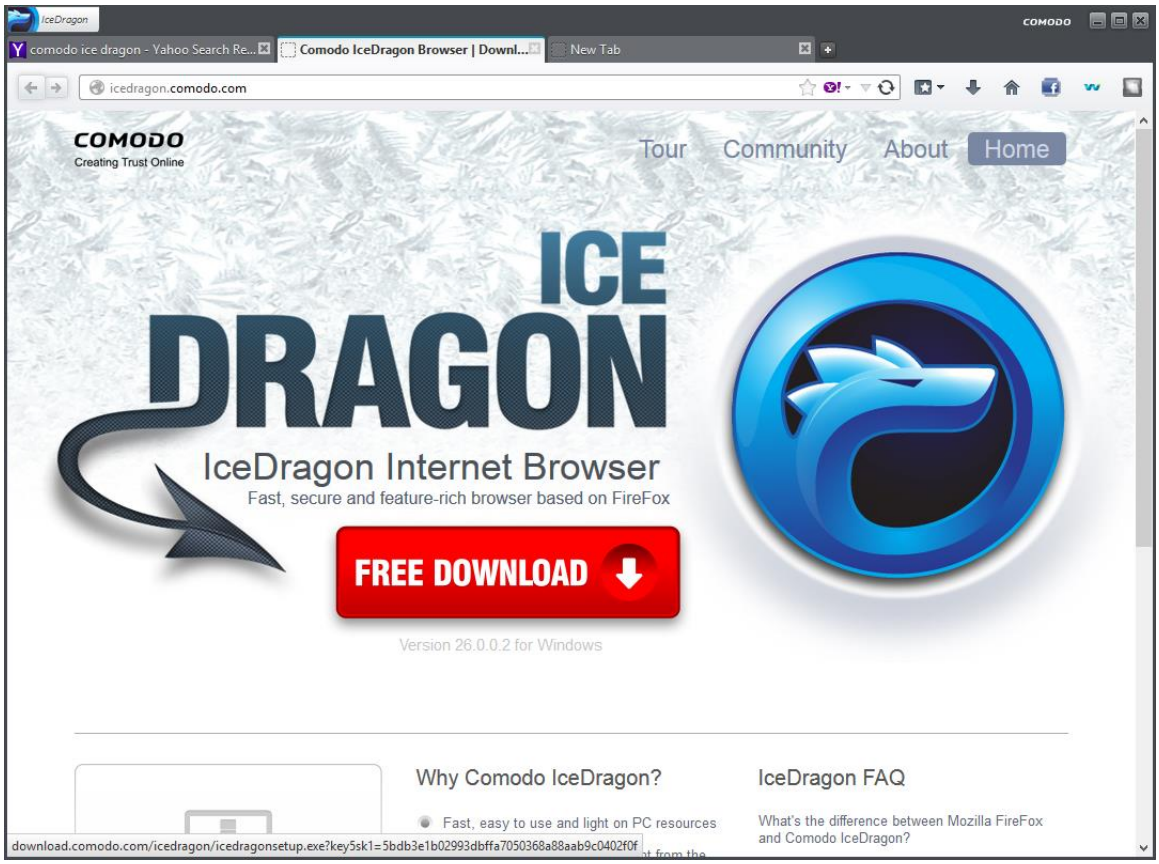
Εικόνα 15 - Ο περιηγητής Wyzo

4.13 Comodo Dragon/Ice Dragon

Η Comodo προσφέρει δύο φυλλομετρητές για ασφαλή περιήγηση στο διαδίκτυο. Έναν βασισμένο στον Google Chrome (Dragon) και έναν βασισμένο στον Mozilla Firefox (Ice Dragon). Στην πραγματικότητα πρόκειται για τροποποιημένες εκδόσεις των δύο πολύ γνωστών περιηγητών με ενισχυμένα χαρακτηριστικά. Και από τους δύο (Firefox και Chrome) έχουν αφαιρεθεί ανεπιθύμητα χαρακτηριστικά ενώ έχει διατηρηθεί η συμβατότητα με τις διάφορες επεκτάσεις του καθενός.

Δίνεται η δυνατότητα στον χρήστη να χρησιμοποιήσει τους Comodo SecureDNS servers τόσο για τον Dragon/Ice Dragon όσο και για οποιαδήποτε άλλη εφαρμογή, που δυνητικά προσφέρει ασφάλεια και ιδιωτικότητα σε σχέση με τον πάροχο internet του κάθε χρήστη.

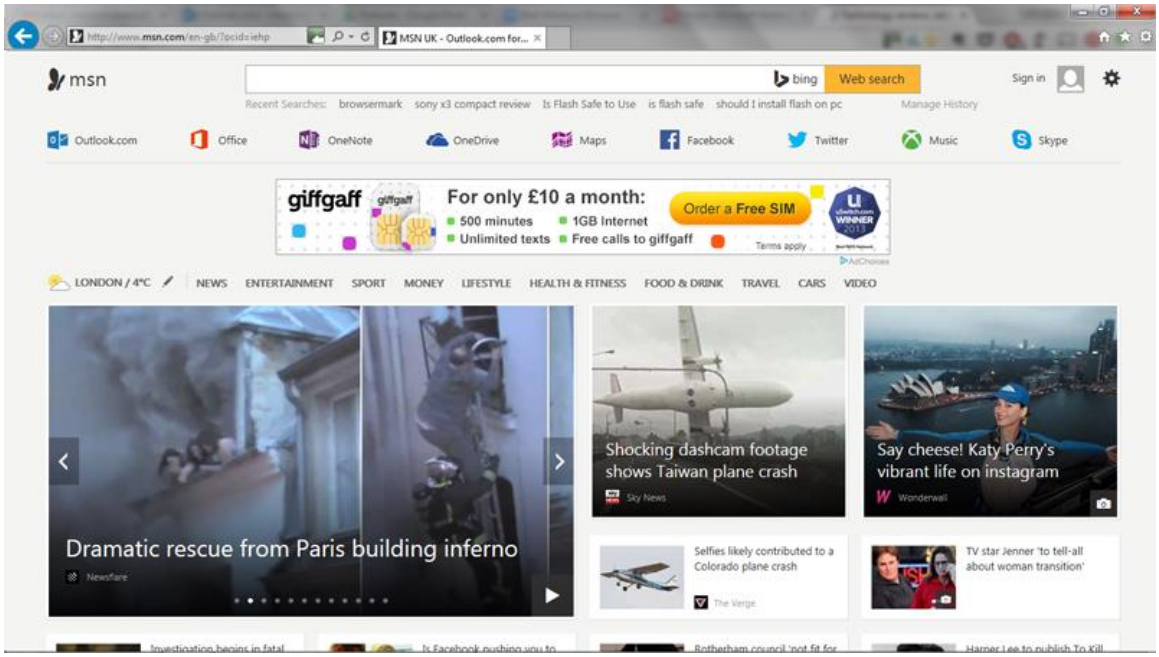
Πιθανώς η πιο ενδιαφέρουσα λειτουργία που υποστηρίζουν είναι η «εικονική λειτουργία» (virtualized mode), κατά την οποία ο περιηγητής απομονώνεται από το υπόλοιπο σύστημα. Το χαρακτηριστικό αυτό διατίθεται εντελώς δωρεάν, με μόνη απαίτηση να υπάρχει εγκατεστημένο στο σύστημα το Comodo Internet Security (CIS), η δωρεάν έκδοση του προγράμματος προστασίας λογισμικού (antivirus) της εταιρίας.



Εικόνα 16 - Ο περιηγητής Ice Dragon

4.14 Microsoft Internet Explorer / Edge

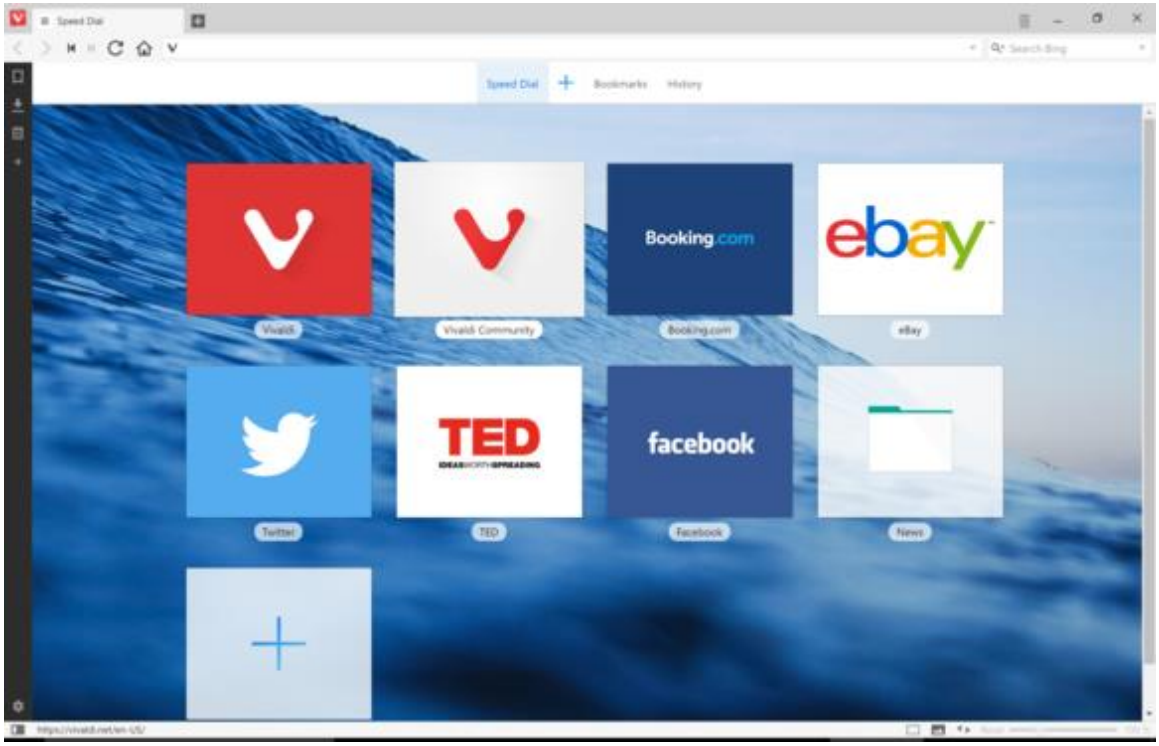
Ο πιο γνωστός και χρησιμοποιημένος browser στον κόσμο είναι ο Internet Explorer. Προσφέρεται δωρεάν από την Microsoft και η τελευταία έκδοση συνοδεύεται από πολλές νέες λειτουργίες. Το μεγαλύτερο μειονέκτημα του είναι ότι μίας και εγκατασταθεί γίνεται κομμάτι του λειτουργικού συστήματος, γεγονός το οποίο κάνει τον υπολογιστή γενικά πιο ευάλωτο στους κινδύνους που εγκυμονεί το διαδίκτυο. Η ιδιωτικότητα είναι κάτι που ενδιαφέρει όλους τους χρήστες τα τελευταία χρόνια και ο Internet Explorer έχει μερικά πολύ χρήσιμα εργαλεία που βοηθούν τον χρήστη να παραμείνει ανώνυμος στο διαδίκτυο. Όπως σχεδόν όλοι οι περιηγητές τα τελευταία χρόνια, έχει την InPrivate λειτουργία κατά την οποία δεν αποθηκεύει ιστορικό περιήγησης ή cookies από τις σελίδες που ο χρήστης επισκέπτεται. Επιπλέον, έχει το χαρακτηριστικό Do Not Track προ-ενεργοποιημένο μαζί με μια αρκετά δυνατή λίστα προστασίας (Tracking Protection List) για να εξασφαλίσει ότι τα δεδομένα του χρήστη δεν θα «πέσουν» σε λάθος χέρια. Ακόμη ένα εργαλείο που έχει εισαχθεί στον Internet Explorer είναι στο Smartscreen. Το Smartscreen αναλύει τους ιστότοπους που επισκέπτεται ο χρήστης και τον προειδοποιεί για τυχόν κακόβουλο περιεχόμενο.



Εικόνα 17 - Ο γνωστός σε όλους περιηγητής Internet Explorer

4.15 Vivaldi

Ένας νέος περιηγητής διαδικτύου που δημιουργήθηκε από προγραμματιστές που αποχώρησαν από την ομάδα ανάπτυξης του Opera. Πρόκειται για έναν φυλλομετρητή που συνδυάζει το παρουσιαστικό του Opera με την λειτουργικότητα του Chrome. Παρέχει χαρακτηριστικά όπως το tab stacking, το οποίο επιτρέπει την δημιουργία ομάδων από καρτέλες. Προσφέρει επίσης τα “Notes”, τη δυνατότητα δηλαδή, να αποθηκεύσει κάποιος ένα μικρό κομμάτι κειμένου, τον ιστότοπο από τον οποίο προήλθε και μια – δυο εικόνες. Προσφέρει επίσης το εργαλείο «Web panels». Ένα web panel είναι ένα μικρό κάθετο sidebar στο οποίο ο χρήστης μπορεί να ρυθμίσει να εμφανίζεται κάποιο τμήμα μιας ιστοσελίδας, πχ ένα Twitter feed. Ο Vivaldi πολύ έξυπνα, ζητά από την ορισμένη σελίδα να επιστρέψει την έκδοση για κινητά τηλέφωνα που είναι ιδανική για περιορισμένους χώρους όπως το συγκεκριμένο sidebar. Τέλος, ο Vivaldi, προσφέρει μια ιδιότητα «content blocker». Το χαρακτηριστικό αυτό είναι κάτι σαν ad blocker με την διαφορά ότι δεν εμποδίζει τις διαφημίσεις, αλλά τις αναγνωρίζει σε μια σελίδα και τις αποκρύπτει από τον χρήστη. Επιπλέον, λόγω του ότι ο Vivaldi έχει βασιστεί στο Chromium, δίνει την δυνατότητα στον χρήστη να εγκαταστήσει επεκτάσεις μέσω του Chrome web store.



Εικόνα 18 - Ο νέος περιηγητής Vivaldi

5. Τύποι επικίνδυνου περιεχομένου

5.1 Drive-By Downloads Επιθέσεις

Μια επίθεση drive-by-download είναι μια τεχνική κατεβάσματος κάποιου κακόβουλου λογισμικού / ιού / shell code, η οποία ενεργοποιείται την στιγμή που ο χρήστης επισκέπτεται μια ιστοσελίδα. Οι επιθέσεις drive-by-download εκτελούνται όταν προσπελαύνεται μια σελίδα ή κάποιο HTML μήνυμα ηλεκτρονικού ταχυδρομείου, το οποίο εκχύει κακόβουλο λογισμικό στον υπολογιστή του ανυποψίαστου θύματος. Αυτού του είδους οι επιθέσεις, συνήθως, κατεβαίνουν και τρέχουν στο παρασκήνιο, έτσι δεν είναι ορατές από τον χρήστη, χαρακτηρίζοντας τες ως μια από τις μεγαλύτερες απειλές στο διαδίκτυο. Τον Απρίλιο του 2007, οι ερευνητές της Google ανακάλυψαν εκατοντάδες χιλιάδες ιστοσελίδες που ενεργοποιούσαν τέτοιες επιθέσεις. Μία στις δέκα ιστοσελίδες βρέθηκε να είναι ύποπτη. Αξιοσημείωτη είναι η παρατήρηση που έγινε από τους ερευνητές του Sophos το 2008 και ανέφερε χαρακτηριστικά ότι 6000 νέες μολυσμένες σελίδες ανακαλύπτονταν κάθε μέρα ή αλλιώς μία σελίδα κάθε 14 δευτερόλεπτα.

5.1.1 Τρόποι αντιμετώπισης

Servers

Στο λειτουργικό σύστημα του server θα πρέπει να είναι εγκατεστημένες οι τελευταίες ενημερώσεις. Όλα τα εγκατεστημένα προγράμματα να είναι στην τελευταία έκδοση. Το μηχάνημα δεν πρέπει να χρησιμοποιείται για περιήγηση στο διαδίκτυο ή ανάγνωση emails. Επίσης πρέπει να γίνει εγγραφή του ιστότοπου, που φιλοξενεί ο server, στα Bing webmaster και Google Webmaster, έτσι ώστε οι μηχανές αναζήτησης να προενεργήσουν εάν αναγνωρίσουν κάτι περίεργο

Τερματικά

Το λειτουργικό σύστημα του υπολογιστή καθώς και ο φυλλομετρητής θα πρέπει να είναι πλήρως ενημερωμένα. Επίσης, πρέπει να γίνεται χρήση κάποιου καλού προγράμματος προστασίας, το οποίο και αυτό με την σειρά του να είναι πλήρως ενημερωμένο. Καλό είναι να αποφεύγονται οι πολλές επεκτάσεις στους διάφορους περιηγητές αφού πολύ συχνά παραβιάζονται, ενώ σε περίπτωση χρήσης του Internet Explorer να είναι ενεργή η επιλογή SmartScreen.

5.2 Clickjacking Επιθέσεις

Οι clickjacking επιθέσεις χρησιμοποιούνται από τους επιτιθέμενους με σκοπό την συλλογή των «κλικς» των χρηστών. Οι επιθέσεις αυτές επιτρέπουν, για λογαριασμό του χρήστη, να εκτελεστεί κάποια ενέργεια στην ιστοσελίδα.

Η ιδέα είναι απλή:

Ο επισκέπτης παρασύρεται με κάποιον τρόπο σε μια ευάλωτη σε clickjacking επιθέσεις σελίδα. Η ιστοσελίδα περιέχει κάποιον σύνδεσμο με κείμενο που θα δελεάσει τον χρήστη, πχ «Γίνε πλούσιος ΤΩΡΑ!!» και z-index=-1 (στον κώδικα). Στη συνέχεια, η σελίδα εντάσσει στον κώδικά της κάποιο «διάφανο» iframe από κάποιο γνωστό domain, πχ facebook.com και το τοποθετεί έτσι ώστε το κουμπί “Like” να βρίσκεται ακριβώς πάνω από τον σύνδεσμο.

Το ακόλουθο κομμάτι κώδικα παρουσιάζει μια τυπική clickjacking επίθεση:

```
1  <style>
2      iframe { /*facebook iframe*/
3          width: 300px;
4          height: 100px;
5          position: absolute;
6          top: 0;
7          left: 0;
8          filter: alpha (opacity=50);
9          /* in real life opacity=0 */ opacity: 0.5;
10     }
11 </style>
12  <div> Click on the link to get rich now: </div>
13  <iframe  src="/files/tutorial/window/clicktarget.html"></iframe>
14 <a href="http://www.google.gr" traget="_blank" style="position : relative; left: 20px; z-index=-1;">
15     CLICK ME
16 </a>
17  <div>You'll be rich for the whole life! </div>
```

Εικόνα 19 - Κώδικας Clickjacking Επίθεσης

5.2.1 Προστασία

Client-side

NoScript

Η επέκταση NoScript για τον Mozilla Firefox, προστατεύει κατά τις Clickjacking επιθέσεις. Η δυνατότητα «ClearClick», η οποία κοινοποιήθηκε τον Οκτώβριο του 2008, προστατεύει τους χρήστες από το προσπελάσουν «αόρατα» ή «καμουφλαρισμένα» αντικείμενα σε κάποια ιστοσελίδα. Σύμφωνα με το «Browser Security Handbook» της Google από το 2008, το ClearClick του NoScript είναι το «μόνο δωρεάν πρόγραμμα που προσφέρει αξιόλογο βαθμό προστασίας» κατά των Clickjacking επιθέσεων.

GuardedID

Το GuardedID (εμπορικό προϊόν) παρέχει προστασία κατά του clickjacking για χρήστες Internet Explorer και Firefox, χωρίς να παρεμβαίνει στην λειτουργία των «νόμιμων» frames της ιστοσελίδας. Το GuardedID επιβάλλει σε όλα τα frames μιας ιστοσελίδας να είναι ορατά.

Gazelle

Ο Gazelle είναι ένας ασφαλής φυλλομετρητής, κατασκευασμένος από την Microsoft, βασισμένος στον Internet Explorer. Χρησιμοποιεί ένα μοντέλο ασφάλειας αντίστοιχο αυτού του λειτουργικού συστήματος και έχει τις δικές του άμυνες κατά των clickjacking επιθέσεων.

Server-side

Framekiller

Οι κάτοχοι και οι διαχειριστές ιστοσελίδων μπορούν να προστατέψουν τους χρήστες/επισκέπτες ενάντια στις clickjacking επιθέσεις, προσθέτοντας στην ιστοσελίδα ένα framekiller. Τα framekillers είναι μια τεχνική που χρησιμοποιείται στις web εφαρμογές με σκοπό να μην προβάλλεται η ιστοσελίδα τους μέσω κάποιου frame. Παρακάτω φαίνεται ένα τυπικό παράδειγμα framekiller:

```
1  <style> html{display:none;} </style>
2  <script>
3     if(self == top) {
4        document.documentElement.style.display = 'block';
5     } else {
6        top.location = self.location;
7     }
8  </script>
```

Εικόνα 20 - Απλό Παράδειγμα Framekiller

Βέβαια μια τέτοιου είδους προστασία, δεν είναι πάντα αξιόπιστη, ειδικά στον Internet Explorer, στον οποίο το αντίμετρο μπορεί να καταστρατηγηθεί από τον ίδιο τον σχεδιασμό του φυλλομετρητή μέσω απλά ενός:

```
1  <iframe security="restricted"></iframe>
2
```

Εικόνα 21 - Αντίμετρο Framekiller σε IE

X-Frame-Options

Το 2009 ο Internet Explorer 8 εισήγαγε έναν νέο HTTP Header, τον X-Frame-Options, οποίος προσέφερε κάποια προστασία κατά των clickjacking επιθέσεων και υιοθετήθηκε και από άλλους φυλλομετρητές. Ο συγκεκριμένος header παίρνει τιμές όπως DENY, SAMEORIGIN και ALLOW-FROM origin, οι οποίες σημαίνουν ότι η ιστοσελίδα απαγορεύει εντελώς την τοποθέτησή της μέσα σε κάποιο frame (DENY), απαγορεύει την τοποθέτησή της μέσα σε frame σε εξωτερική σελίδα ή επιτρέπει την τοποθέτησή της σε κάποιο frame σε συγκεκριμένη ιστοσελίδα. Επιπλέον, μπορεί να πάρει την τιμή ALLOWALL, στην οποία επιτρέπεται το framing του περιεχομένου σε οποιαδήποτε άλλη σελίδα.

Content Security Policy

Η ντιρεκτίβα frame-ancestors της πολιτικής ασφάλειας περιεχομένου έχει την δυνατότητα να επιτρέψει ή να απαγορεύσει την προσθήκη περιεχομένου με χρήση iframe, object κ.α. από πιθανώς εχθρικές σελίδες. Η συγκεκριμένη ντιρεκτίβα, καθιστά τον HTTP-Header, X-Frame-Options, ξεπερασμένο. Σε περίπτωση που μια ιστοσελίδα παρέχεται και με τους δύο headers, ο φυλλομετρητής προτιμά την πολιτική του frame-ancestors.

Παράδειγμα πολιτικών frame-ancestors:

```
# Disallow embedding. All iframes etc. will be blank,
# or contain a browser specific error page. Content-Security-Policy: frame-ancestors 'none'
# Allow embedding of [[same-origin policy|own content]] only.
Content-Security-Policy: frame-ancestors 'self'
# Allow specific origins to embed this content
Content-Security-Policy: frame-ancestors example.com wikipedia.org
```

Εικόνα 22 - Πολιτική Frame-Ancestors

Plug-ins και Script-Enabled Επιθέσεις

Οι επιτιθέμενοι στοχεύουν πρόσθετα στους φυλλομετρητές ή εκχύουν κακόβουλο κώδικα, συνήθως Javascript, σε διαδικτυακές εφαρμογές. Αυτού του είδους οι επιθέσεις βοηθούν στην επίτευξη drive-by-download ή Clickjacking επιθέσεων.

5.3 Επιθέσεις Phishing

Phishing είναι η προσπάθεια εξαπάτησης κάποιου χρήστη, συνήθως μέσω email που εμφανίζονται να προέρχονται από νόμιμους αποστολείς (πανεπιστήμια, τράπεζες κτλ.), με σκοπό να υποκλέψουν προσωπικά στοιχεία του χρήστη. Τα μηνύματα αυτά, συνήθως οδηγούν σε πλαστές σελίδες ή πείθουν τον χρήστη να συμπληρώσει κάποια φόρμα με προσωπικά δεδομένα όπως κωδικούς πρόσβασης, αριθμούς πιστωτικών καρτών κ.α. Οι επιτιθέμενοι χρησιμοποιούν τις πληροφορίες αυτές για πλαστογράφηση ταυτότητας. Τα phishing emails πάντα ζητούν από το χρήστη να ακολουθήσει κάποιο σύνδεσμο, που οδηγεί σε μια ιστοσελίδα με φόρμα προς συμπλήρωση. Οι Νόμιμοι Οργανισμοί δεν ζητούν ποτέ τέτοιου είδους πληροφορία μέσω email.

5.4 Social (Engineering) Network Επιθέσεις

Το social engineering είναι η τέχνη της χειραγώγησης του κοινού με σκοπό την αποκάλυψη απόρρητων πληροφοριών. Το είδος των πληροφοριών που αναζητείται από τους επιτιθέμενους ποικίλει, όταν όμως ο στόχος είναι κάποιος μεμονωμένος χρήστης, τότε προσπαθούν να του αποσπάσουν κωδικούς πρόσβασης, τραπεζικούς λογαριασμούς ή κάποια απομακρυσμένη πρόσβαση στον προσωπικό υπολογιστή τους. Η τεχνική είναι πολύ δημοφιλής στους επιτιθέμενους και χρησιμοποιείται διότι είναι πιο εύκολο να πείσουν έναν χρήστη να δώσει τον κωδικό πρόσβασής του παρά να τον μαντέψουν οι ίδιοι. Υπάρχουν 2 τύποι επιθέσεων social-engineering:

5.4.1 Human Based

Αυτοί οι τύποι επιθέσεων social engineering χρειάζονται ανθρώπινη αλληλεπίδραση, κατά πρόσωπο επαφή δηλαδή και έπειτα την ανάκτηση των επιθυμητών πληροφοριών.

5.4.2 Computer Based

Οι επιθέσεις social-engineering βασισμένες σε υπολογιστή, χρησιμοποιούν λογισμικό που επιχειρεί να ανακτήσει την επιθυμητή πληροφορία. Πχ. Phishing, Baiting, On-line απάτες.

5.4.3 Προστασία

Προσεχτικός έλεγχος των emails

Ένα phishing email μήνυμα μπορεί να ισχυρίζεται ότι προέρχεται από κάποια «νόμιμη» εταιρία και όταν ακολουθηθεί ο σύνδεσμος που αναφέρει, εμφανίζεται μια ολόιδια ιστοσελίδα με την πραγματική. Το email μπορεί να ζητά από το θύμα να δώσει προσωπικές πληροφορίες αλλά συνήθως δεν αναφέρει το όνομα του θύματος. Τα περισσότερα phishing emails ξεκινάνε με το «Dear Customer» («Αγαπητέ Πελάτη»). Μια νόμιμη εταιρία δεν θα έστελνε spam emails και θα απευθυνόταν στον πελάτη της με ολόκληρο το όνομα του.

Μην εισάγετε Οικονομικές ή προσωπικές πληροφορίες

Τα περισσότερα από τα phishing emails οδηγούν τα θύματα σε ιστοσελίδες οι οποίες απαιτούν την καταχώρηση οικονομικών ή προσωπικών πληροφοριών. Ένας χρήστης του διαδικτύου δεν πρέπει ποτέ να αποκαλύπτει εμπιστευτικές πληροφορίες μέσω συνδέσμων που προέρχονται από emails.

Αναγνώριση ψεύτικου τηλεφωνήματος

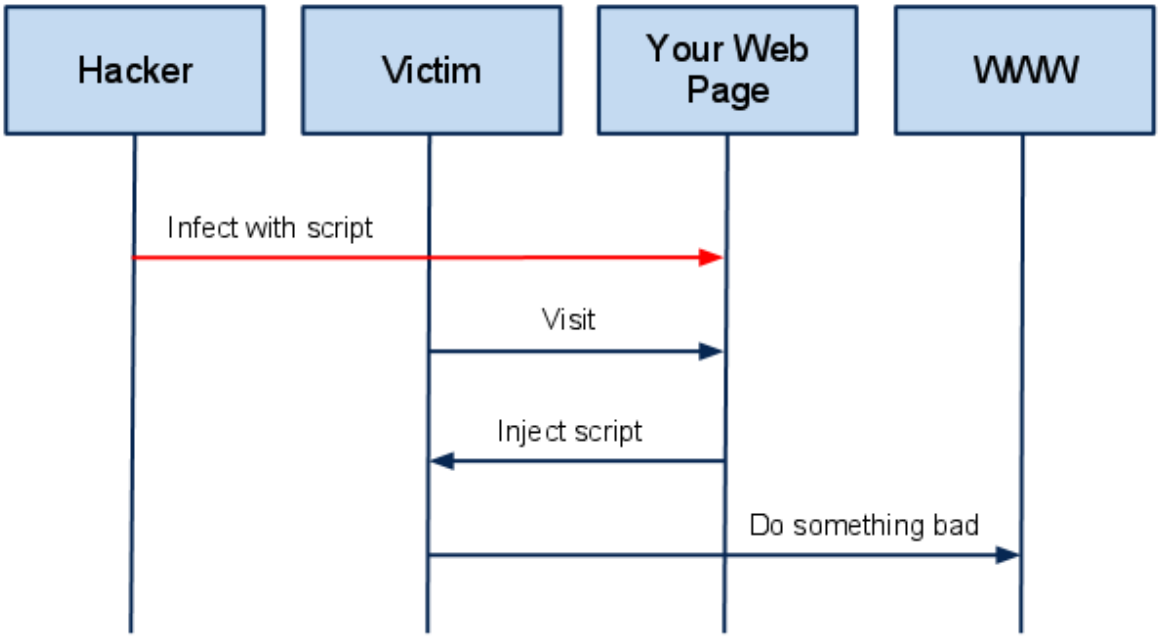
Το phone phishing είναι μια μέθοδος κατά την οποία ο επιτιθέμενος προσπαθεί να αποσπάσει πληροφορίες από το θύμα, μέσω τηλεφωνικής κλήσης. Πχ. ο χρήστης μπορεί να κληθεί να παράσχει οικονομικά στοιχεία για να λάβει υποτιθέμενη επιστροφή χρημάτων σε κάποιον λογαριασμό του. Το τηλεφώνημα μπορεί να προέρχεται από έναν αριθμό ο οποίος φαίνεται θεμιτός, όμως ο κωδικός της περιοχής στο εν λόγο τηλεφώνημα μπορεί να τροποποιηθεί μέσω VOIP.

Προστασία μέσω Λογισμικού

Anti-spyware και firewall λογισμικά μπορούν να χρησιμοποιηθούν για την αποτροπή επιθέσεων τύπου phishing. Οι χρήστες θα πρέπει να κρατούν, κάθε στιγμή, ενημερωμένα τα προγράμματα και επιπλέον, όπως προαναφέρθηκε, να μην αποστέλλουν προσωπικά στοιχεία μέσω email.

5.5 Cross Site Scripting (XSS) Επιθέσεις

Το Cross-site scripting γνωστό και ως XSS επίθεση είναι μια τεχνική hacking στο επίπεδο της εφαρμογής. Σε μια XSS επίθεση, ο hacker μολύνει κάποια ιστοσελίδα με client-side κώδικα. Όταν ο χρήστης επισκέπτεται την σελίδα, το script κατεβαίνει στον φυλλομετρητή του και εκτελείται. Οι επιθέσεις XSS ακολουθούν το ακόλουθο διάγραμμα.



Εικόνα 23 - Διάγραμμα ροής ενός XSS Attack

5.5.1 Non-Persistent XSS Attack

Απαιτούν να επισκεφτεί ένας χρήστης κάποιο συγκεκριμένο σύνδεσμο, δημιουργημένο από κάποιον επιτιθέμενο. Όταν ο χρήστης ακολουθήσει αυτόν τον σύνδεσμο, ο κακόβουλος κώδικας θα εκτελεστεί στον φυλλομετρητή του χρήστη. Το ακόλουθο script είναι ένα τυπικό παράδειγμα non-persistent XSS επίθεσης:

```
index.php :
<?php
    $name = $_GET['name'];
    echo "Welcome $name <br />";
    echo "<a href=/"http://xssattackexamples.com/">Click to Download</a>";
?>

call to index.php:
index.php?name=guest<script>alert('attacked');</script>
```

Εικόνα 24 - Τυπικό Non persistent XSS attack

5.5.2 Persistent XSS Attack

Σε αυτόν τον τύπο επίθεσης, ο κώδικας εκχύεται από τον επιτιθέμενο και αποθηκεύεται σε μια βάση δεδομένων. Η βλάβη που προκαλείται από αυτή την επίθεση είναι πιο σοβαρή σε σχέση με την Non-Persistent.

5.5.3 Προστασία

Ο OWASP μας δίνει 13 κανόνες για να αποφευχθεί μια XSS επίθεση:

1. Αποφυγή εισαγωγής εμπιστευτικών δεδομένων παρά μόνο σε συγκεκριμένα σημεία
2. Κωδικοποίηση (escaping) των δεδομένων σαν HTML κώδικα πριν την εισαγωγή τους σε κάποιο HTML αντικείμενο
3. Κωδικοποίηση (escaping) Χαρακτηριστικών
4. Κωδικοποίηση (escaping) JavaScript
5. Κωδικοποίηση (escaping) JSON
6. Κωδικοποίηση (escaping) CSS
7. Κωδικοποίηση (escaping) URL
8. Καθαρισμός του HTML κώδικα με χρήση βιβλιοθήκης σχεδιασμένης για αυτή τη δουλειά
9. Αποφυγή DOM-based XSS
10. Χρήση του HTTPOnly cookie flag
11. Υλοποίηση Πολιτικής Ασφάλειας Περιεχομένου (Content Security Policy)
12. Χρήση συστήματος αυτόματης κωδικοποίησης
13. Χρήση του X-XSS-Protection Response Header

5.6 SQL Injection Attacks

Το sql injection είναι μια τεχνική, με την οποία οι επιτιθέμενοι μπορούν να προσθέσουν εντολές SQL, μέσω κάποιας διαδικτυακής φόρμας, σε κάποιο υπάρχον SQL ερώτημα. Οι εντολές αυτές αλλάζουν το αρχικό ερώτημα και μπορούν να θέσουν σε κίνδυνο την ασφάλεια της δικτυακής εφαρμογής. Η αρχική μορφή μιας SQL Injection επίθεσης είναι η εισαγωγή κώδικα σε μεταβλητές που ουσιαστικά δέχονται είσοδο από τον χρήστη και ενώνονται με άλλες εντολές SQL για να εκτελεστούν. Μια άλλη μορφή SQL Injection είναι η εισαγωγή του κακόβουλου κώδικα σε συμβολοσειρές οι οποίες αποθηκεύονται σε κάποια βάση δεδομένων. Όταν οι αποθηκευμένες συμβολοσειρές συνενώνονται σε μια δυναμική εντολή SQL, ο κακόβουλος κώδικας εκτελείται. Το ακόλουθο παράδειγμα μας δείχνει μια απλή επίθεση SQL Injection:

```
3  var Shipcity;
4  ShipCity = Request.form ("ShipCity");
5  var sql = "select * from OrdersTable where ShipCity = '" + ShipCity + "'";
```

Εικόνα 25 - PHP κώδικας ευπαθής σε SQL Injection

Ζητείται από τον χρήστη να εισάγει το όνομα της πόλης. Αν βάλει "Athens", το ερώτημα διαμορφώνεται από το script ως:

```
3  SELECT * FROM OrdersTable WHERE ShipCity = 'Athens'
```

Εικόνα 26 - Ερώτημα ευπαθές σε SQL Injection

Το script χτίζει ένα SQL ερώτημα από την συνένωση συγκεκριμένων συμβολοσειρών με την τιμή που εισήγαγε ο χρήστης.

5.6.1 Αντιμετώπιση

Prepared (Parameterized) Statements

Τα prepared/parameterized statements είναι ανθεκτικά ενάντια σε επιθέσεις SQL Injection, καθώς οι παράμετροι του ερωτήματος, οι οποίες μεταδίδονται σε δεύτερο χρόνο χρησιμοποιώντας διαφορετικό πρωτόκολλο δεν χρειάζεται να γίνουν «escape». Αν το αρχικό πρότυπο ερώτημα δεν προέρχεται από εξωτερική είσοδο, δεν μπορεί να επιτευχθεί επίθεση SQL Injection.

Stored Procedures

Οι παράμετροι στις stored procedures θα αντιμετωπιστούν ως δεδομένα ακόμα και στην περίπτωση που ο επιτιθέμενος εισάγει εντολές SQL. Επίσης, κάποιες βάσεις δεδομένων κάνουν έλεγχο του τύπου των παραμέτρων. Παρόλα αυτά, μια stored procedure που δημιουργεί δυναμικά SQL χρησιμοποιώντας την είσοδο του χρήστη, είναι ακόμα ευάλωτη σε SQL Injection επιθέσεις.

Least Privilege Principle (Αρχή του μικρότερου προνομίου)

Γνωστή και ως principle of minimal privilege ή principle of least authority, προβλέπει πως κάθε μέρος ενός υπολογιστικού συστήματος θα πρέπει να έχει πρόσβαση μόνο στις πληροφορίες και στους πόρους που είναι απολύτως απαραίτητα για την λειτουργία του.

Επαλήθευση εισόδου με λίστα επιτρεπόμενων (White List Input Validation)

Η επαλήθευση εισόδου μπορεί να χρησιμοποιηθεί για την ανίχνευση κακόβουλης εισόδου πριν από την επεξεργασία της από την εφαρμογή. Η επαλήθευση εισόδου με λίστα επιτρεπόμενων απαιτεί τον ακριβή ορισμό της εισόδου που επιτρέπεται και εξ ’ορισμού οτιδήποτε δεν ανήκει στην λίστα αυτή, απορρίπτεται.

5.7 Third-Party Web Apps Επιθέσεις

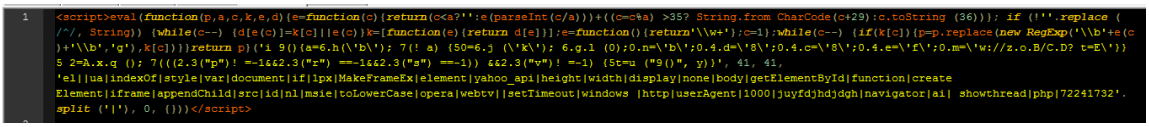
Όσες περισσότερες third-party δικτυακές εφαρμογές χρησιμοποιούν σε εταιρικό περιβάλλον οι υπάλληλοι μιας επιχείρησης, τόσες περισσότερες πιθανότητες έχει κάποιος επιτιθέμενος να χρησιμοποιήσει κάποια από αυτές τις εφαρμογές για να εισβάλλει στο εταιρικό δίκτυο ή και στα τερματικά μηχανήματα της εταιρίας. Σε περίπτωση που δεν είναι γνωστή ή δεν υπάρχει κάποια πολιτική στην επιχείρηση για τις εφαρμογές αυτές, τότε είναι αναμενόμενο πως οι χρήστες θα εγκαταστήσουν κάποιες στα εταιρικά μηχανήματα. Σε περίπτωση επίθεσης μέσω του εσωτερικού δικτύου της επιχείρησης, ο επιτιθέμενος θα έστελνε κάποιο email που θα περιείχε κάποιο συνημμένο αρχείο με τον κακόβουλο κώδικα. Στην περίπτωση δικτυακών εφαρμογών, η επίθεση θα απαιτούσε να διαμοιραστεί κάποιο αρχείο μέσω κάποιας νόμιμης εφαρμογής όπως Dropbox, Box, SalesForce.com ή Google Drive.

5.8 JavaScript Obfuscation Επιθέσεις

Το Obfuscation είναι μια τεχνική για την απόκρυψη των επιθέσεων από εργαλεία στατικής ανάλυσης, τα οποία χρησιμοποιούν υπογραφές για να συγκρίνουν με γνωστούς κακόβουλους κώδικες. Το Obfuscation αλλάζει τον κώδικα έτσι ώστε να αποφεύγει τα εργαλεία ανίχνευσης κακόβουλου λογισμικού. Η Javascript είναι μια δυναμική γλώσσα προγραμματισμού, την οποία

χρησιμοποιούν οι επιτιθέμενοι για να αναπτύξουν κακόβουλο κώδικα, ο οποίος θα εισαχθεί σε μια "καθαρή" ιστοσελίδα.

Το ακόλουθο παράδειγμα δείχνει το obfuscation ενός Javascript κώδικα:



Εικόνα 27 - Obfuscated Javascript

Ο παραπάνω ασαφής κώδικας, προέρχεται από τον ακόλουθο Javascript κώδικα:

```
function MakeFrameEx ()
{
    element = document.getElementById ('yahoo_api');
    if (! element) {
        var el = document.createElement ('iframe');
        document.body.appendChild (el);
        el.id = 'yahoo_api';
        el.style.width = '1px';
        el.style.height = '1px';
        el.style.display = 'none';
        el.src = 'hxxp: //juyfdjhdjdgh.nl.ai/showthread.php?t? T=72241732'
    }
}
var ua = navigator.userAgent.toLowerCase ();
if (((ua.indexOf ("msie")! =- 1 && ua.indexOf ("opera") ==- 1 && ua.indexOf ("webtv")==-1)) && ua.indexOf ("windows")! =- 1) {
    var t = setTimeout ("MakeFrameEx ()", 1000)
}
```

Εικόνα 28 - Πραγματικός (De-obfuscated) κώδικας Javascript

6. Google Chrome Extensions

Ο Google Chrome είναι ίσως ο πιο διαδεδομένος περιηγητής διαδικτύου αυτήν την στιγμή, με εκδόσεις για όλες τις πλατφόρμες (PC, Mac, Linux, Mobile), φανατικούς χρήστες και μεγάλη γκάμα επεκτάσεων. Συγχρονίζει σε όλες τις πλατφόρμες και εκτελεί το ίδιο, τα plugins που ο χρήστης έχει εγκαταστήσει. Είναι μια καλή λύση για να είναι η εφαρμογή μας cross-platform. Να λειτουργεί το ίδιο δηλαδή, χωρίς να χρειάζεται να ξαναγραφτεί σε κάποια άλλη γλώσσα προγραμματισμού (native source code), ούτε να εξαρτάται η ορθή λειτουργία της από το εκάστοτε λειτουργικό σύστημα. Η Google παρέχει το δικό της API και κατευθύνσεις για τον σωστό τρόπο συγγραφής ενός chrome-extension. Μέσω αυτού του API μας δίνονται πολλές δυνατότητες αλληλεπίδρασης με τον ίδιο τον browser αλλά και με τις ιστοσελίδες που επισκέπτεται ο χρήστης.

6.1 Αρχιτεκτονική Chrome Extension

Όπως αναφέρεται στο "overview" της Google, μία επέκταση (extension) είναι ένας συμπιεσμένος φάκελος από αρχεία, HTML, Javascript, css, εικόνες και οτιδήποτε άλλο χρειάζεται που προσθέτει λειτουργικότητα στον browser. Ουσιαστικά οι επεκτάσεις είναι ιστοσελίδες που χρησιμοποιούν το API που παρέχει ο browser σε μια σελίδα. Τα extensions αλληλεπιδρούν με τις ιστοσελίδες χρησιμοποιώντας content-scripts ή cross origin XMLHttpRequests.

Βασικά Αρχεία Επεκτάσεων

- Manifest.json
- Html files
- Javascript files

6.1.1 Manifest.json

Δίνει πληροφορίες για την επέκταση όπως τα σημαντικά αρχεία και δυνατότητες που μπορεί να χρησιμοποιήσει.

```
{
  "manifest_version": 2,

  "name": "Getting started example",
  "description": "This extension shows a Google Image search result for the current page",
  "version": "1.0",

  "browser_action": {
    "default_icon": "icon.png",
    "default_popup": "popup.html"
  },
  "permissions": [
    "activeTab",
    "https://ajax.googleapis.com/"
  ]
}
```

Εικόνα 29 - Απλό Παράδειγμα αρχείου Manifest.json

6.1.2 Background Page

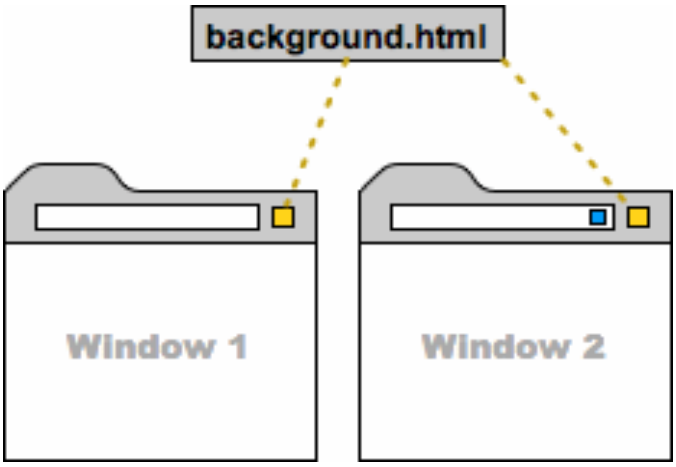
Πολλά extensions έχουν μια background page, μια "αόρατη" σελίδα που έχει την λογική (business logic) του extension. Τα background pages χωρίζονται σε 2 είδη:

Event Pages

Φορτώνονται μόνο όταν χρειάζονται. όταν δεν είναι ενεργά, ελευθερώνουν την μνήμη και όλους τους υπόλοιπους πόρους που είχαν δεσμεύσει.

Background pages

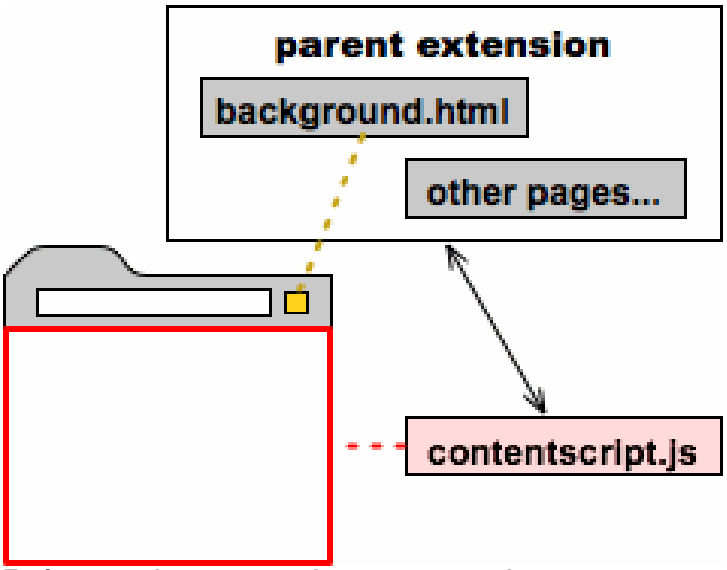
Πρόκειται για μια background σελίδα (HTML) η οποία "τρέχει" μέσα στην διεργασία του extension. Υπάρχει μόνο για την περίοδο που «τρέχει» το ίδιο το extension και μόνο ένα αντικείμενό της είναι κάθε φορά ενεργό.



Εικόνα 30 - Chrome Extension - Background σελίδα

6.1.3 Content Scripts

Χρησιμοποιούνται στην περίπτωση που το extension πρέπει να αλληλοεπιδρά με τις ιστοσελίδες. Ένα content-script είναι κάποιος Javascript κώδικας που εκτελείται μέσα στην ίδια σελίδα που έχει φορτωθεί στον browser. Είναι ουσιαστικά μέρος της επισκεπτόμενης σελίδας και όχι μέρος του extension. Τα content scripts έχουν την δυνατότητα να διαβάζουν λεπτομέρειες από τις ιστοσελίδες καθώς και να κάνουν αλλαγές σε αυτές.



Εικόνα 31 - Chrome Extension - Content Script

7. Docker

7.1 Τι είναι το Docker

Το Docker είναι μια πλατφόρμα λογισμικού ανοιχτού κώδικα που υλοποιεί Εικονικοποίηση (Virtualization) σε επίπεδο Λειτουργικού Συστήματος. Ουσιαστικά το Docker προσφέρει αυτοματοποιημένες διαδικασίες για την ανάπτυξη εφαρμογών σε απομονωμένες Περιοχές Χρήστη (User Spaces) που ονομάζονται Software Containers. Το λογισμικό χρησιμοποιεί τεχνολογίες του πυρήνα των Linux όπως τα cgroups και οι χώροι ονομάτων πυρήνα (kernel namespaces), για να επιτρέπει σε ανεξάρτητα software containers να εκτελούνται στο ίδιο λειτουργικό σύστημα. Με τον τρόπο αυτό αποφεύγεται η χρήση των επιπλέον υπολογιστικών πόρων που θα απαιτούσε μια εικονική μηχανή (virtual machine).

Το Docker είναι ένα εργαλείο που μπορεί να ενοποιήσει μια εφαρμογή μαζί με τις εξαρτήσεις της σε ένα εικονικό περιβάλλον (container) καθιστώντας το ικανό να «τρέξει» σε οποιονδήποτε server. Αυτό βοηθά στη φορητότητα της εφαρμογής, αφού μπορεί να τρέξει στις εγκαταστάσεις κάποιας εταιρίας, σε κάποιο δημόσιο cloud δίκτυο, σε ιδιωτικό cloud κ.α.

Το Docker παρέχει «ελαφριά» εικονικά περιβάλλοντα που τρέχουν διεργασίες σε απομόνωση, υλοποιώντας ένα υψηλού επιπέδου API.

Επειδή χρησιμοποιεί θεμέλια που παρέχει ο πυρήνας των Linux, ένα Docker container δεν απαιτεί ούτε περιλαμβάνει ένα ξεχωριστό Λειτουργικό Σύστημα, αντίθετα με μια εικονική μηχανή. Αντ' αυτού, στηρίζεται στην λειτουργικότητα του πυρήνα και χρησιμοποιεί μια τεχνική απομόνωσης

πόρων και ξεχωριστές περιοχές (namespaces) για να απομονώσει την εικόνα του Λειτουργικού Συστήματος στην εφαρμογή.

Χρησιμοποιώντας τα containers, οι πόροι μπορούν να απομονωθούν, οι υπηρεσίες να περιοριστούν και οι διεργασίες μπορούν να έχουν σχεδόν μια εντελώς ιδιωτική εικόνα του Λειτουργικού Συστήματος, της δομής των αρχείων και των διεπαφών δικτύων. Πολλαπλά containers μοιράζονται τον ίδιο πυρήνα, αλλά κάθε container μπορεί να περιοριστεί στο να χρησιμοποιεί συγκεκριμένους πόρους όπως CPU, μνήμη και I/O.

Η χρήση του Docker για την δημιουργία και διαχείριση containers μπορεί να απλοποιήσει την δημιουργία κατανεμημένων συστημάτων, αφού επιτρέπει να τρέχουν ταυτόχρονα πολλαπλές εφαρμογές και διάφορες άλλες διεργασίες, σε ένα μόνο φυσικό μηχάνημα ή σε πολλαπλά εικονικά.

Το Docker είναι ένα βασικό εργαλείο σαν την java και το git, που μπορεί να ενταχθεί στην καθημερινότητα του κύκλου ανάπτυξης μιας εφαρμογής. Μπορεί να χρησιμοποιηθεί και να διευκολύνει μια ομάδα ανάπτυξης σε διάφορες διαδικασίες. Μπορεί να χρησιμοποιηθεί ως σύστημα ελέγχου εκδόσεων (version control system) για ολόκληρο το Λειτουργικό Σύστημα της εφαρμογής. Είναι μια πολύ καλή λύση για την περίπτωση που μια ομάδα θέλει να διαμοιραστεί το Λειτουργικό Σύστημα μιας εφαρμογής, όπως επίσης και να εκτελεστεί η εφαρμογή σε οποιονδήποτε προσωπικό υπολογιστή στο ίδιο ακριβώς περιβάλλον που εκτελείται και στον server.

7.1.1 Το Docker είναι όπως η Java

Η Java έχει δώσει μια μεγάλη υπόσχεση: *Γράψτε μια φορά. Εκτελέστε οπουδήποτε.*

Το Docker υπόσχεται ακριβώς το ίδιο. Με εξαίρεση ίσως τον κώδικα, μπορεί κάποιος να ρυθμίσει τον server ακριβώς όπως θέλει (διαλέγοντας Λειτουργικό Σύστημα, βελτιστοποιώντας αρχεία ρυθμίσεων της εφαρμογής, εγκαθιστώντας εκτελέσιμα κ.α.) και να είναι σίγουρος ότι η πρότυπη εικόνα του server θα τρέξει σε οποιοδήποτε Docker server με τον ίδιο ακριβώς τρόπο, παράγοντας τα ίδια ακριβώς αποτελέσματα.

Για παράδειγμα στην Java θα είχαμε το παρακάτω κομμάτι κώδικα:

```
1  /**
2  *    File : HelloWorld.java
3  */
4
5  class HelloWorldApp {
6      public static void main (string[] args){
7          System.out.println("Hello World.");
8      }
9  }
10
```

Εικόνα 32 - Τυπικό HelloWord σε Java

Στη συνέχεια θα εκτελούσαμε την εντολή :

javac HelloWorld.java

με την οποία θα «χτίζαμε» το πρόγραμμά μας. Η εντολή αυτή σαν αποτέλεσμα θα είχε την παραγωγή του εκτελέσιμου αρχείου *HelloWorld.class*, το οποίο μπορεί να εκτελεστεί σε οποιοδήποτε μηχάνημα που έχει εγκατεστημένη Εικονική Μηχανή Java (JVM).

Αντιστοίχως στο Docker θα είχαμε ένα *Dockerfile* αρχείο:

```
C:\Users\spyridon.liagkos\Desktop\test.js
##

##

FROM ubuntu:13.10

ENV DEBIAN_FRONTEND noninteractive

RUN apt-get update -qq -y && apt-get install curl -qq -y && apt-get clean

RUN curl -sSL https://get.rvm.io | bash -s stable --ruby=2.1.1
```

Εικόνα 33 - Απλό Dockerfile αρχείο

Για το «χτίσιμο» θα χρησιμοποιούσαμε την εντολή: *docker build -t my/ruby*, η οποία θα δημιουργούσε ένα container με όνομα «*my/ruby»* το οποίο και θα μπορούσε να εκτελεστεί σε οποιοδήποτε μηχάνημα με Docker server.

Η ιδιότητα αυτή του Docker το καθιστά ιδανική λύση για το server της εφαρμογής μας. Θέλοντας η εφαρμογή να λειτουργεί ανεξάρτητα από το λειτουργικό σύστημα, το Docker αποτελεί την καλύτερη λύση σε σχέση ταχύτητας – όγκου – ευκολίας εγκατάστασης.

7.1.2 Το Docker είναι όπως το git

Το git, υπόσχεται ότι: Ελάχιστο αποτύπωμα με αστραπιαία απόδοση (Tiny footprint with lightning fast performance.).

Όπως και με την Java, το Docker υπόσχεται το ίδιο με μόνη διαφορά ότι δεν παρακολουθεί αλλαγές σε κώδικα αλλά σε συστήματα. Το git έχοντας δυνατότητες όπως την δημιουργία τοπικών κλαδιών του κώδικα, πολλαπλές ροές εργασίας κ.α., ξεπερνά κατά πολύ τα υπόλοιπα εργαλεία διαχείρισης κώδικα όπως το Subversion, το CVS, το Perforce και το ClearCase. Έτσι και το Docker ξεπερνά τα υπόλοιπα εργαλεία διαχείρισης containers, παρέχοντας απίστευτα γρήγορους χρόνους εκκίνησης (microseconds και όχι λεπτά), ευκολόχρηστα εργαλεία χτισίματος εικόνων κ.α.

Για παράδειγμα στο Git μπορεί κάποιος να δει τις αλλαγές που έχουν γίνει με την εντολή *git status*:

```
1  git init .
2  touch README.md
3  git add .
4  git status
5  On branch master
6  Initial commit
7
```

Εικόνα 34 - git status εντολή

Αλλαγές που πρόκειται να γίνουν commit:
Προσθήκη *README.md* αρχείου στο git

```
1  git commit -am "Adding README.md"
2  [master (root-commit) 78184aa] Adding README.md
3  1 file changed, 0 insertions(+), 0 deletions(-)
4  create mode 100644 README.md
5
```

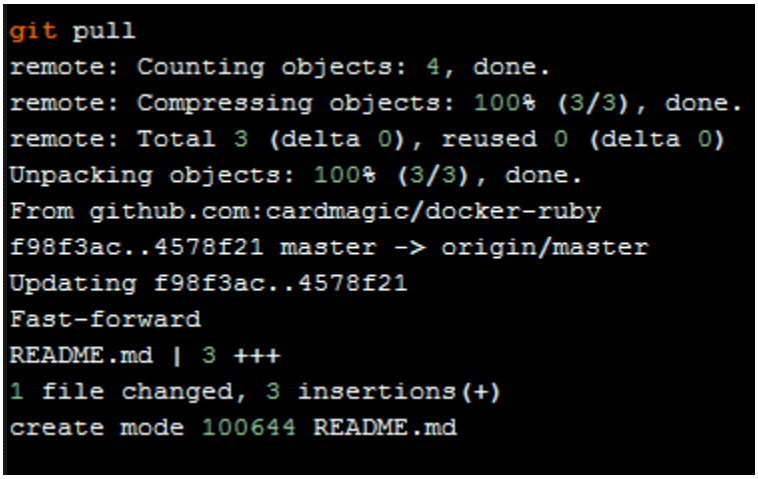
Εικόνα 35 - Εντολή git commit

«Ανέβασμα» των αλλαγών στο git repository

```
git push
Counting objects: 49, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (39/39), done.
Writing objects: 100% (49/49), 4.29 KiB | 0 bytes/s, done.
Total 49 (delta 13), reused 0 (delta 0)
To git@github.com:my/repo.git
* [new branch] master -> master
```

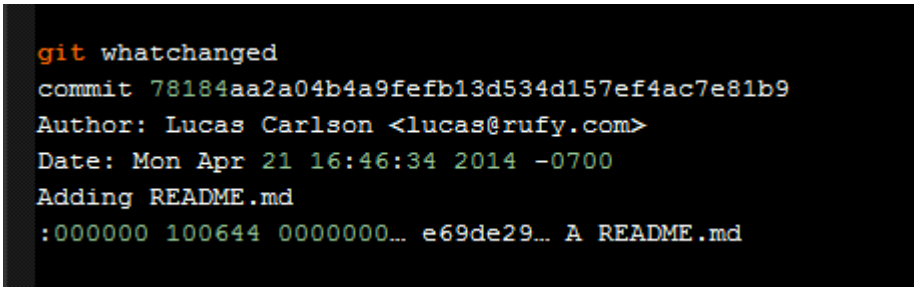
Εικόνα 36 - Εντολή git push

Η τοπική διακλάδωση με όνομα «master» έχει ρυθμιστεί να παρακολουθεί την διακλάδωση του server με όνομα «master»



Εικόνα 37 - Εντολή git pull

Χρήση της εντολής *whatchanged* για την προβολή των αλλαγών



Εικόνα 38 - Εντολή git whatchanged

Αντίστοιχα στο Docker μπορεί ο χρήστης να παρακολουθήσει τις αλλαγές που έχουν γίνει σε ολόκληρο το σύστημα:

Εικόνα 39 - Εντολή docker diff

Commit εντολή (αντίστοιχη του Git)

```
docker commit -m "Adding README.md" $MY_DOCKER my/ubuntu
4d46072299621b8e5409cbc5d325d5ce825f788517101fe63f5bda448c9954da
```

Εικόνα 40 - Εντολή docker commit

Εντολή «ανεβάσματος» του Docker για την αναβάθμιση του repository:

```
docker push my/ubuntu
The push refers to a repository [my/ubuntu] (len: 1)
Sending image list
Pushing repository my/ubuntu (1 tags)
511136ea3c5a: Image already pushed, skipping
Image 6170bb7b0ad1 already pushed, skipping
Image 9cd978db300e already pushed, skipping
de2fdfc8f7d8: Image successfully pushed
Pushing tag for rev [de2fdfc8f7d8] on {https://registry-1.docker.io/v1/repositories/my/ubuntu/tags/latest}
```

Εικόνα 41 - Εντολή docker push

Εντολή «κατεβάσματος» του Docker για την λήψη του container:

```
docker pull my/ubuntu
Pulling repository my/ubuntu
de2fdfc8f7d8: Download complete
511136ea3c5a: Download complete
6170bb7b0ad1: Download complete
9cd978db300e: Download complete
```

Εικόνα 42 - Εντολή docker pull

Εντολή ιστορικού αλλαγών στο Docker:

```
docker history my/ubuntu
IMAGE CREATED CREATED BY SIZE
de2fdfc8f7d8 3 minutes ago bash -c touch README.md; sleep 10000 77 B
9cd978db300e 11 weeks ago /bin/sh -c #(nop) ADD precise.tar.xz in / 204.4 MB
6170bb7b0ad1 11 weeks ago /bin/sh -c #(nop) MAINTAINER Tianon Gravi
```

Εικόνα 43 - Εντολή docker history

8. WebNinja Application

8.1 Web-Ninja chrome extension

Η εφαρμογή μας (Web-Ninja) αποτελείται από 1 manifest.json αρχείο για την δήλωση των δυνατοτήτων και εξαρτήσεων της επέκτασης, 1 background script για την "λογική" καθώς και 1 content script για την αλληλεπίδραση της επέκτασης με την σελίδα που προσπελαύνει ο χρήστης.

Το background script είναι 1 σύνολο από συναρτήσεις, διαδικασίες και event-handlers. Αναγνωρίζει το πάτημα στο κουμπί του extension καθώς και την ανανέωση της ιστοσελίδας. Μόλις αντιληφθεί κάποια ανανέωση σε σελίδα ή το πάτημα του action-button, τότε εκτελεί το content script του extension, έτσι ώστε να μπορεί να αλληλοεπιδράσει με την ιστοσελίδα στην οποία πλοηγείται ο χρήστης.

Οι handlers:

- chrome.tabs.onUpdated
- chrome.browserAction.onClicked

αναγνωρίζουν την ανανέωση και το πάτημα του κουμπιού του extension αντίστοιχα. Αφού αναγνωρισθεί 1 από τα 2 events, τότε το extension διαλέγει από την λίστα των αντίστοιχων καρτελών αυτή που είναι ενεργή εκείνη την στιγμή και εκτελεί για την συγκεκριμένη καρτέλα το content script.

Το content script με την σειρά του, εκχύει στον αρχικό κώδικα της ιστοσελίδας στην οποία περιηγείται ο χρήστης, την βιβλιοθήκη jQuery και 1 μικρό Javascript κώδικα. η βιβλιοθήκη jQuery δεν είναι αναγκαία αλλά βοηθάει στον χειρισμό της σελίδας αλλά και στην εύκολη κλήση ajax προς τον server της εφαρμογής. Το μικρό κομμάτι κώδικα που επιπλέον εκχύει στον αρχικό κώδικα της σελίδας, για αρχή "αδειάζει" το περιεχόμενο της σελίδας. Με αυτό επιτυγχάνει να μην πέσει ο χρήστης θύμα κάποιας drive-by-download επίθεσης, αφού ο πρωτότυπος κώδικας Javascript της ιστοσελίδας, δεν προλαβαίνει να εκτελεστεί.

Στη συνέχεια, αφού σβηστεί το αρχικό περιεχόμενο της σελίδας, αναγνωρίζει το url (διεύθυνση) της ιστοσελίδας και εκτελεί μια HTTP POST κλήση προς τον server στον οποίο επικοινωνεί το url. Ο server απαντά με 1 object όπως το παρακάτω:

```
{
    head: "",
    body: ""
}
```

Το αντικείμενο αυτό, φέρει τον καθαρό κώδικα της σελίδας που ο χρήστης προσπαθεί να προσπελάσει. Με τον όρο καθαρό εννοούμε πως η σελίδα έχει ελεγχθεί και από τον πηγαίο κώδικά της έχει αφαιρεθεί οποιοδήποτε τμήμα κώδικα θεωρήθηκε κακόβουλο ή ύποπτο καθώς και οποιοδήποτε τμήμα κώδικα της σελίδας έφερε κάποια διαφήμιση.

8.2 Server side

Το **Node.js** είναι μια πλατφόρμα ανάπτυξης λογισμικού (κυρίως διακομιστών) χτισμένη σε περιβάλλον Javascript. Στόχος του Node είναι να παρέχει ένα εύκολο τρόπο δημιουργίας κλιμακωτών διαδικτυακών εφαρμογών. Σε αντίθεση από τα περισσότερα σύγχρονα περιβάλλοντα ανάπτυξης εφαρμογών δικτύων μία διεργασία *node* δεν στηρίζεται στην πολυνηματικότητα αλλά σε ένα μοντέλο ασύγχρονης επικοινωνίας εισόδου/εξόδου.

Το *Node* χαρακτηρίζεται από την έμφαση στην ασύγχρονη επικοινωνία μεταξύ των υπολογιστικών πόρων. Αυτό επιτυγχάνεται με την χρήση συμβάντων (*events*) που προσφέρει η Javascript και ονομάζονται callbacks. Για παράδειγμα όταν ένας περιηγητής ιστού φορτώσει πλήρως ένα αρχείο, ένας χρήστης πατάει κάποιο κουμπί, ολοκληρώνεται ένα αίτημα AJAX, τα συμβάντα αυτά πυροδοτούν ένα συγκεκριμένο *callback*. Αυτό με την σειρά του επιτρέπει την ροή του κώδικα χωρίς να αφήνει ανενεργό τον επεξεργαστή προκειμένου να εκτελεστεί μια λειτουργία, όπως μια επιτυχής ανάγνωση αρχείου από τον δίσκο.

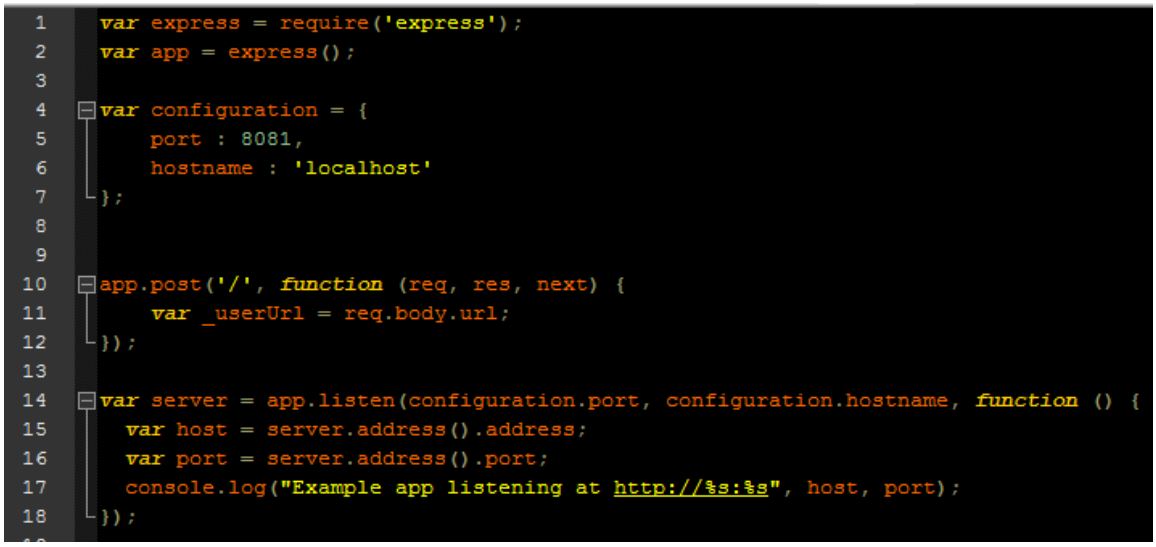
Εκεί που πραγματικά ξεχωρίζει το Node.js είναι στην δημιουργία γρήγορων, κλιμακωτών δικτυακών εφαρμογών αφού είναι ικανό να χειριστεί τεράστιο αριθμό ταυτόχρονων συνδέσεων με μεγάλο throughput.

Ο τρόπος με τον οποίο λειτουργεί είναι αρκετά ενδιαφέρον. Αντίθετα με τις παραδοσιακές τεχνικές δικτυακής εξυπηρέτησης, στις οποίες κάθε σύνδεση δημιουργεί μια νέα διεργασία, που καταλαμβάνει μνήμη από το σύστημα μέχρι να μην μείνει άλλη διαθέσιμη, το Node.js τρέχει σε 1 μονάχα διεργασία χρησιμοποιώντας ένα μοντέλο ασύγχρονης επικοινωνίας εισόδου/εξόδου. Αυτός ο τρόπος του επιτρέπει να υποστηρίζει δεκάδες χιλιάδες ταυτόχρονες συνδέσεις. Με ένα γρήγορο υπολογισμό: Υποθέτοντας ότι σε 1 μηχάνημα με συνολικά 8 GB RAM, κάθε νήμα διεργασίας καταλαμβάνει δυνητικά 2MB μνήμης RAM τότε έχουμε 1 θεωρητικό μέγιστο σύνολο 4000 ταυτόχρονων συνδέσεων και μην ξεχνάμε και το κόστος της εναλλαγής μεταξύ των νημάτων. Με την αποφυγή όλων αυτών, το Node.js επιτυγχάνει την διαχείριση περισσότερων από 1000000 ταυτόχρονες συνδέσεις.

Η ανθεκτικότητα στον μεγάλο αριθμό συνδέσεων καθιστά το Node.js ιδανική πλατφόρμα για την ανάπτυξη της εφαρμογής από την πλευρά του εξυπηρετητή. Ακόμα 1 μεγάλο πλεονέκτημα είναι η μεγάλη "γκάμα" βιβλιοθηκών που υπάρχει στην εν λόγω πλατφόρμα και βοηθούν στην καλύτερη διαχείριση των διεργασιών αλλά και των δεδομένων.

Μια από τις πλέον διαδεδομένες πλατφόρμες, βασισμένη στο Node.js, είναι το Express.js. Το Express.js ή απλά Express, είναι μια πλατφόρμα ανάπτυξης διαδικτυακών εφαρμογών για τον Node.js. Είναι σχεδιασμένο για το χτίσιμο διαδικτυακών εφαρμογών και APIs. Εκ των πραγμάτων πρόκειται για την πρότυπη πλατφόρμα διακομιστών για Node.js, το οποίο το καθιστά ιδανικό για την εφαρμογή μας, αφού η μοναδική της απαίτηση είναι να υπάρχει 1 REST API στο οποίο η client μεριά να αποστέλλει τα προς έλεγχο δεδομένα.

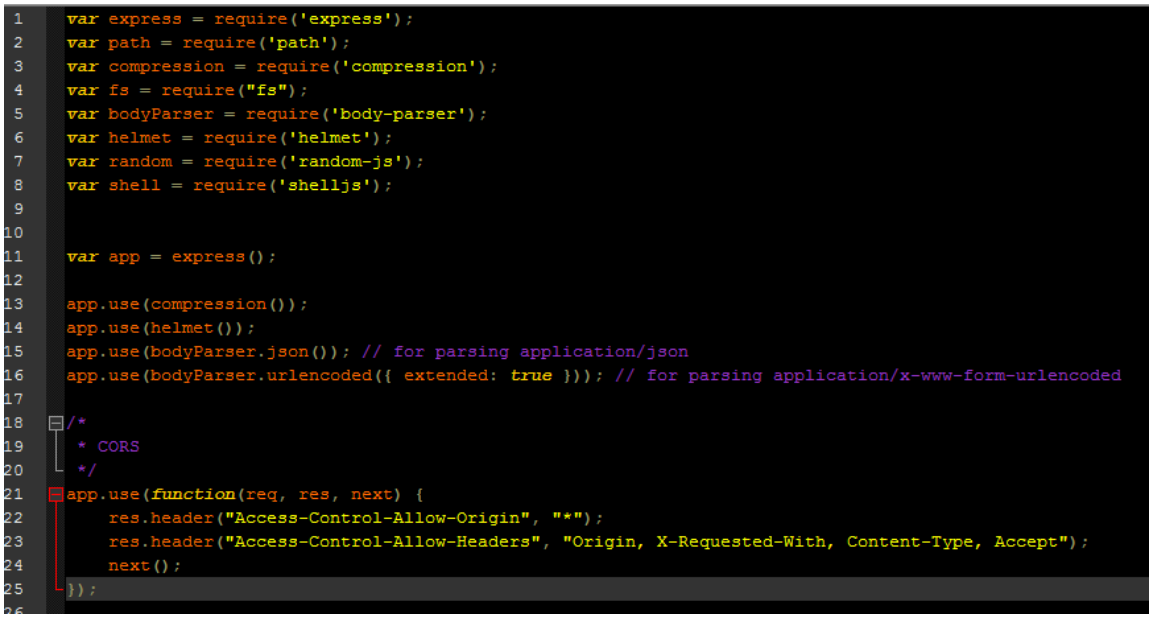
Ξεκινώντας λοιπόν, δημιουργούμε μέσω express έναν απλό εξυπηρετητή, που δέχεται μία HTTP Post κλήση.



```
var express = require('express');
var app = express();

var configuration = {
    port : 8081,
    hostname : 'localhost'
};


app.post('/', function (req, res, next) {
    var _userUrl = req.body.url;
});

var server = app.listen(configuration.port, configuration.hostname, function () {
  var host = server.address().address;
  var port = server.address().port;
  console.log("Example app listening at http://%s:%s", host, port);
});
```

Εικόνα 44 - Webninja Http server

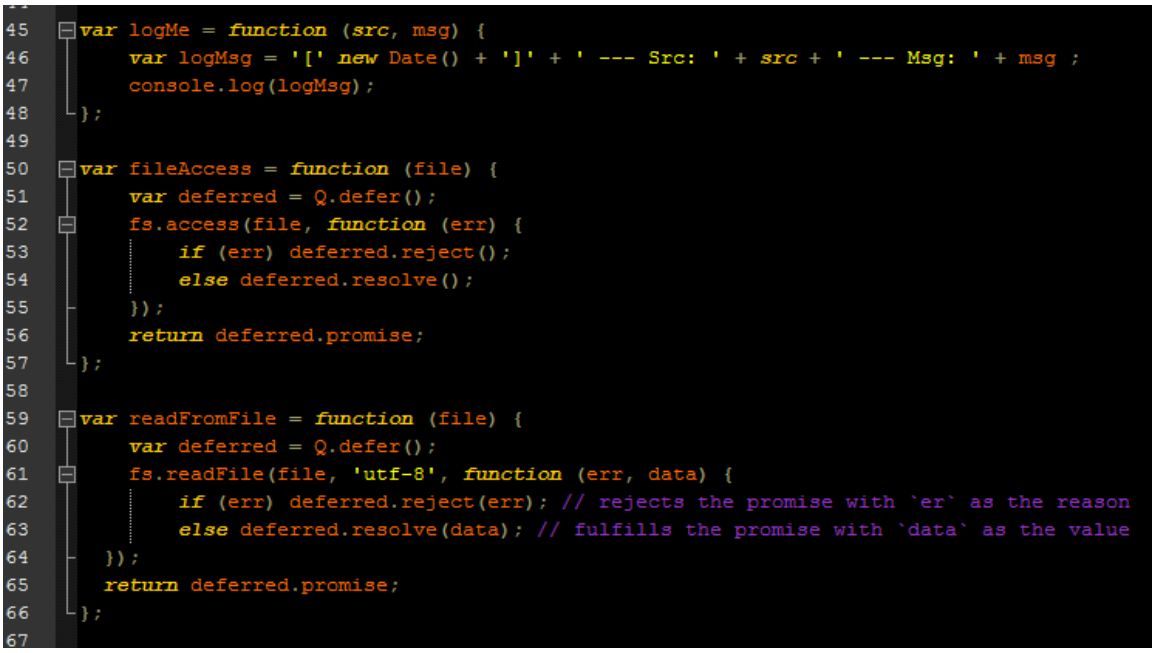
Τώρα που έχουμε δημιουργήσει 1 βασικό web server, θα τον εμπλουτίσουμε με στοιχεία που θα χρησιμεύσουν στην εφαρμογή μας όπως: δυνατότητα να αλληλοεπιδρά με το λειτουργικό σύστημα και το filesystem, να παράγει τυχαίους αριθμούς, να αναλύει το σώμα των HTTP κλήσεων και να συμπιέζει τα δεδομένα που αποστέλλει στον client. Αυτές τις δυνατότητες θα τις πετύχουμε προσθέτοντας στο script μας εξαρτήσεις σε επεκτάσεις/βιβλιοθήκες του Express.



```
var express = require('express');
var path = require('path');
var compression = require('compression');
var fs = require("fs");
var bodyParser = require('body-parser');
var helmet = require('helmet');
var random = require('random-js');
var shell = require('shelljs');


var app = express();

app.use(compression());
app.use(helmet());
app.use(bodyParser.json()); // for parsing application/json
app.use(bodyParser.urlencoded({ extended: true })); // for parsing application/x-www-form-urlencoded

/*
 * CORS
 */
app.use(function(req, res, next) {
    res.header("Access-Control-Allow-Origin", "*");
    res.header("Access-Control-Allow-Headers", "Origin, X-Requested-With, Content-Type, Accept");
    next();
});
```

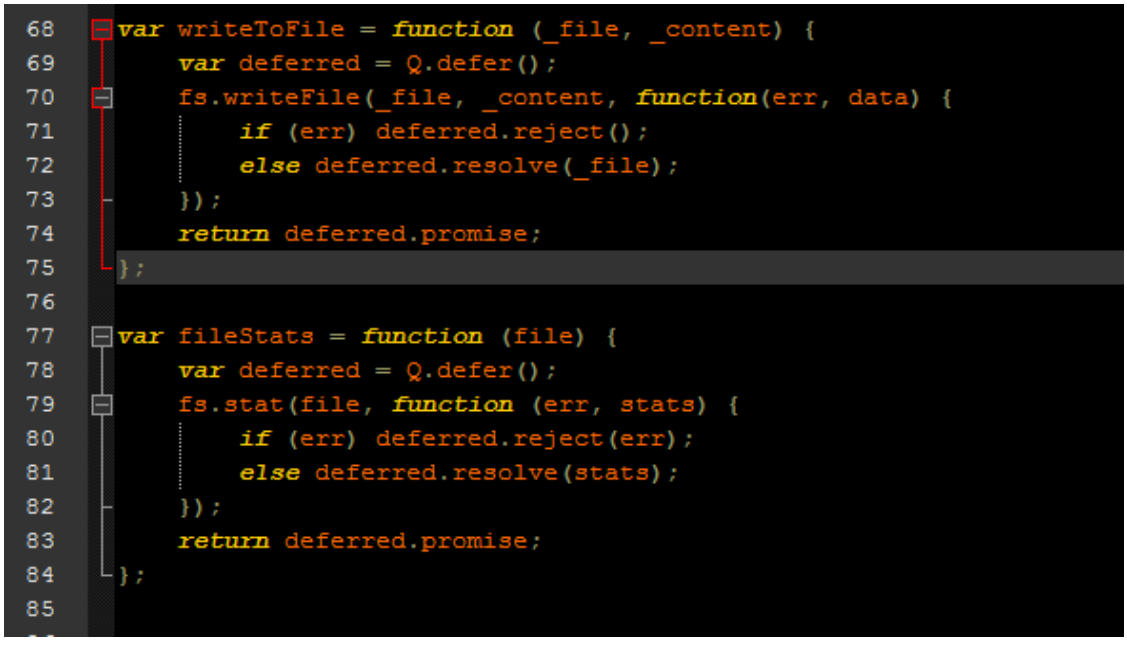
Εικόνα 45 - Εξαρτήσεις και βοηθητικές βιβλιοθήκες του Webninja

Στη συνέχεια θα προσθέσουμε κάποιες γενικές συναρτήσεις που θα χρησιμεύσουν, όπως logging, διάβασμα και γράψιμο αρχείων, έλεγχος ύπαρξης αρχείου στο δίσκο.



```
45  var logMe = function (src, msg) {
46      var logMsg = '[' new Date() + ']' + ' --- Src: ' + src + ' --- Msg: ' + msg ;
47      console.log(logMsg);
48  };
49
50  var fileAccess = function (file) {
51      var deferred = Q.defer();
52      fs.access(file, function (err) {
53          if (err) deferred.reject();
54          else deferred.resolve();
55      });
56      return deferred.promise;
57  };
58
59  var readFromFile = function (file) {
60      var deferred = Q.defer();
61      fs.readFile(file, 'utf-8', function (err, data) {
62          if (err) deferred.reject(err); // rejects the promise with `er` as the reason
63          else deferred.resolve(data); // fulfills the promise with `data` as the value
64    });
65    return deferred.promise;
66  };
67
```

Εικόνα 46 - Βοηθητικές συναρτήσης Webninja (μέρος 1)



```
68  var writeToFile = function (_file, _content) {
69      var deferred = Q.defer();
70      fs.writeFile(_file, _content, function(err, data) {
71          if (err) deferred.reject();
72          else deferred.resolve(_file);
73      });
74      return deferred.promise;
75  };
76
77  var fileStats = function (file) {
78      var deferred = Q.defer();
79      fs.stat(file, function (err, stats) {
80          if (err) deferred.reject(err);
81          else deferred.resolve(stats);
82      });
83      return deferred.promise;
84  };
85
```

Εικόνα 47 - Βοηθητικές συναρτήσεις Webninja (μέρος 2)

Τι θα κάνει όμως πραγματικά ο server;

Αλγοριθμικά:

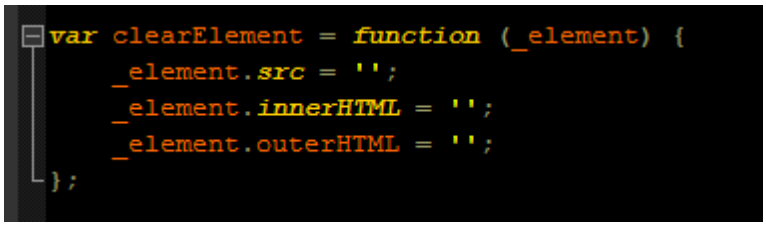
- ✓ Διαβάζει τα δεδομένα που στέλνει ο client.
- ✓ Κατεβάζει, αν δεν υπάρχει ήδη, στον δίσκο μια λίστα με διευθύνσεις που θεωρούνται διαφημίσεις
- ✓ Κατεβάζει τοπικά την σελίδα που προσπαθεί ο χρήστης να προσπελάσει και αφαιρεί από τον κώδικά της οποιοδήποτε script και iframe έχει scr κάτι που να μοιάζει με οποιοδήποτε εγγραφή στην λίστα αποκλεισμού.
- ✓ Σώζει την καθαρή από διαφημίσεις σελίδα σε 1 προσωρινό αρχείο στον δίσκο
- ✓ εκτελεί το JsUnpack-n με είσοδο το αρχείο από το προηγούμενο βήμα
- ✓ καθαρίζει αν υπάρχει λόγος το αρχείο από τα κομμάτια κώδικα που το JsUpack-n χαρακτήρισε επικίνδυνα.
- ✓ επιστρέφει την καθαρή πλέον σελίδα στον χρήστη.

Για να μπορέσει η εφαρμογή να κατεβάσει και να "καθαρίσει" την σελίδα, θα πρέπει να λειτουργεί όπως ένας φυλλομετρητής. Για τον σκοπό αυτό, έχει δημιουργηθεί η βιβλιοθήκη PhantomJS. Στην πραγματικότητα πρόκειται για έναν headless browser, ένα φυλλομετρητή χωρίς γραφικό περιβάλλον χρήστη, που χρησιμοποιείται για την αυτοματοποίηση της αντίδρασης (αλληλεπίδρασης) μιας ιστοσελίδας. Το PhantomJS παρέχει ένα Javascript API, μέσω του οποίου μπορεί κάποιος να εξομοιώσει την περιήγηση στη σελίδα, να πάρει screenshot, να μιμηθεί συμπεριφορές χρηστών ακόμα κ.α. καθιστώντας το ένα συνηθισμένο εργαλείο για να εκτελεί κάποιος τεστ βασισμένα σε φυλλομετρητές μέσω κάποιου συστήματος χωρίς γραφικό περιβάλλον.

Όπως φαίνεται και στο παρακάτω κομμάτι κώδικα (συνάρτηση getClientsPage), μέσω του PhantomJS, η εφαρμογή "κατεβάζει" και κρατά στην μνήμη την σελίδα στην οποία περιηγείται το χρήστης. Στην συνέχεια διατρέχει όλα τα αντικείμενα της σελίδας και ξεχωρίζει όσα είναι τύπου script και iframe. Ελέγχει εάν σε κάποιο από αυτά η τιμή του πεδίου "src" ταιριάζει με κάποια τιμή μέσα από την λίστα αποκλεισμού. Στην περίπτωση που ταιριάζει, τότε "καθαρίζει" το συγκεκριμένο αντικείμενο. Με τον όρο καθαρίζει, εννοούμε ότι σβήνει την τιμή του πεδίου "src" και στην συνέχεια διαγράφει το αντικείμενο από την σελίδα θέτοντας το outerHTML = ''. Αφού διατρέξει και καθαρίσει όλα τα script και iframe αντικείμενα της ιστοσελίδας, επιστρέφει ως αποτέλεσμα τον "καθαρό" HTML κώδικα.

```
phantom.create().then(function (ph) {
    ph.createPage().then(function (page) {
        page.open(_url).then(function (status) {
            page.evaluate(function(regex, clearElement) {
                for (var i=0; i < document.scripts.length; i++ ) {
                    if ((new RegExp(regex)).test(document.scripts[i].src))
                        clearElement(document.scripts[i]);
                }

                for (var i=0; i < document.body.getElementsByTagName('iframe').length; i++) {
                    if ((new RegExp(regex)).test(document.body.getElementsByTagName('iframe')[i].src))
                        clearElement(document.body.getElementsByTagName('iframe')[i]);
                }
                var head = document.head.outerHTML;
                var body = document.body.outerHTML;
                return '<html>' + head + body + '</html>';
            }, regex, clearElement)
            .then(function(html){
                page.close();
                ph.exit();
            });
        });
    });
});
```

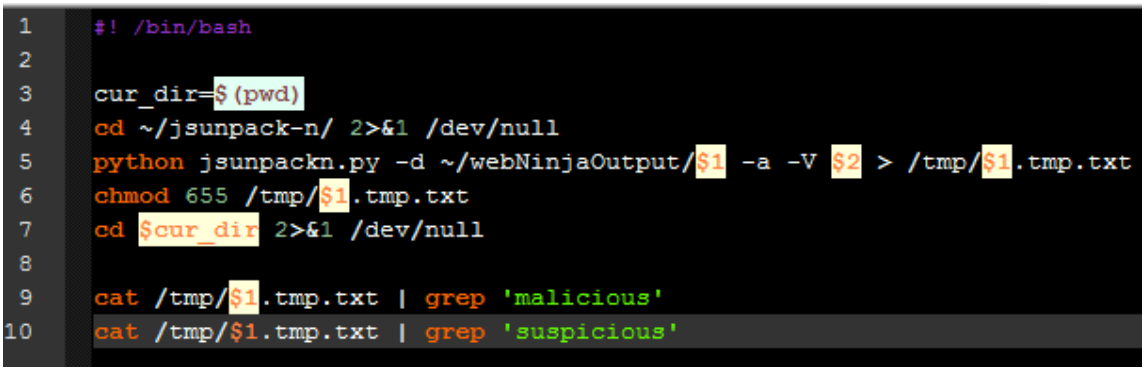
Εικόνα 48 - Σώμα συνάρτησης getClientsPage



```
var clearElement = function (_element) {
    _element.src = '';
    _element.innerHTML = '';
    _element.outerHTML = '';
};
```

Εικόνα 49 - Συνάρτηση clearElement

Μετά την αφαίρεση των διαφημίσεων από την σελίδα, η εφαρμογή αποθηκεύει την σελίδα σε 1 προσωρινό αρχείο, με μοναδικό όνομα, στον δίσκο και ξεκινά την διαδικασία ελέγχου για κακόβουλο λογισμικό. Για τον έλεγχο αυτό, χρησιμοποιεί την command line έκδοση του εργαλείου jsUnpack. Για να αλληλοεπιδράσει το Node.js με το λειτουργικό σύστημα υπάρχει το ενσωματωμένο module "child process". Η εφαρμογή όμως χρησιμοποιεί ένα npm module, το shelljs, αντί του "child process", το οποίο κάνει πιο εύκολη την διαχείριση των διεργασιών καθώς και όλη την αλληλεπίδραση με το λειτουργικό σύστημα. Πρακτικά, η εφαρμογή θα "τρέξει" μέσω του shelljs το ακόλουθο bash script.



```
#! /bin/bash

cur_dir=$(pwd)
cd ~/jsunpack-n/ 2>&1 /dev/null
python jsunpackn.py -d ~/webNinjaOutput/$1 -a -V $2 > /tmp/$1.tmp.txt
chmod 655 /tmp/$1.tmp.txt
cd $cur_dir 2>&1 /dev/null

cat /tmp/$1.tmp.txt | grep 'malicious'
cat /tmp/$1.tmp.txt | grep 'suspicious'
```

Εικόνα 50 - shell script που αναγνωρίζει μια σελίδα ως malicious ή suspicious

Το script μεταφράζεται σε shelljs στον παρακάτω κώδικα με κάποιες παραλλαγές.

```
var tmpFile = shell.tempdir() + "/" + userId + ".tmp.txt";
var cur_dir = shell.pwd();
shell.cd(configuration.jsUnpackDir);
var cmd = 'python jsunpackn.py -d ~/webNinjaOutput/' + userId + " -a -V " + _file;
shell.exec(cmd, {silent:true}, function (code, stdout, stderr) {
    logMe('execJsUnpack', 'shell.exec callback');
    var isClear = true;
    var txtToReturn = '';
    var textToRemove = [];
    if ( code !== 0) {
        deferred.reject(stderr);
    }
    else {
        shell.ShellString(stdout).to(tmpFile);
        shell.chmod(655, tmpFile);
        shell.cd(cur_dir);
        var isMalicious = shell.cat(tmpFile).grep('malicious').stdout.replace(/^.*\[malicious:.*$/mg, '').trim();
        var isSuspicious = shell.cat(tmpFile).grep('suspicious').stdout.replace(/^.*\[suspicious:.*$/mg, '').trim();
        isClear = (isMalicious.length === 0 && isSuspicious.length === 0 );
        txtToReturn = isMalicious + '<br />' + isSuspicious;
        txtToReturn = txtToReturn.replace('\n', '<br />');
    }
    shell.cd(cur_dir);
    deferred.resolve({isClear: isClear, text: txtToReturn, file: _file});
});
```

Εικόνα 51 - Μετάφραση του shell script σε Javascript μέσω shelljs

Πρακτικά ο κώδικας θα αποθηκεύσει σε μια μεταβλητή τον τρέχοντα φάκελο και θα μεταβεί στον φάκελο εγκατάστασης του jsunpack-n. Στην συνέχεια θα εκτελέσει το jsunpack, δίνοντάς του ως παραμέτρους τον φάκελο εξόδου για τα αποτελέσματα (παράμετρος -d), το αρχείο που θα εξετάσει (παράμετρος "_file"), την εντολή να επισκεφτεί και να αναλύσει σε πραγματικό χρόνο τα URL που περιέχονται στο αρχείο (παράμετρος -a) καθώς και να τυπώσει την μέγιστη δυνατή πληροφορία (παράμετρος -V).

Το shelljs εκτελεί ασύγχρονα την εντολή μέσω της συνάρτησης exec και ανακατευθύνει το αποτέλεσμά της στο προσωρινό αρχείο (εντολή shell.ShellString(stdout).to(tmpFile)). Αλλάζει τα δικαιώματα του αρχείου σε 655 έτσι ώστε να μην είναι εκτελέσιμο και επιστρέφει στον αρχικό φάκελο. Στη συνέχεια σαρώνει το προσωρινό αρχείο που αποθήκευσε προηγουμένως και ψάχνει για γραμμές που περιέχουν είτε την λέξη "malicious" είτε την λέξη "suspicious". Εάν δεν βρει γραμμές που να περιέχουν 1 από τις 2 λέξεις, τότε η σελίδα που επισκέπτεται ο χρήστης θεωρείται ασφαλής. Σε αντίθετη περίπτωση, το jsunpack, έχει δημιουργήσει, στον φάκελο εξόδου που

ορίστηκε κατά την κλήση του, ένα αρχείο με ονομασία original_<hash_key>, στο οποίο έχει αποθηκεύσει τον κακόβουλο κώδικα που περιέχει η ιστοσελίδα.

Αφού ολοκληρωθεί η ανάλυση της σελίδας και χαρακτηριστεί «καθαρή», ύποπτη ή επικίνδυνη, η εφαρμογή θα δημιουργήσει την τελική σελίδα που θα δει ο χρήστης. Στην περίπτωση που δεν βρέθηκε κάτι ύποπτο στον ιστότοπο, τότε, η εφαρμογή χρησιμοποιώντας το module jsdom, διαβάζει το αρχείο με τον καθαρό κώδικα της σελίδας και τον μετατρέπει σε αντικείμενο τύπου document element της Javascript. Το jsdom περιέχει μια ασύγχρονη διαδικασία που μέσω του χαρακτηριστικού "file" δέχεται το αρχείο εισαγωγής, αυτό που θα διαβάσει δηλαδή. Στην δική μας περίπτωση το αρχείο με τον καθαρό κώδικα. Μέσω του χαρακτηριστικού "scripts" μπορούν να ορισθούν scripts που θα εκχυθούν στον αυθεντικό κώδικα του αρχείου εισαγωγής. Η εφαρμογή χρησιμοποιεί το χαρακτηριστικό αυτό για να εισάγει στον κώδικα, στο τέλος του, την βιβλιοθήκη jQuery. Τέλος, με το χαρακτηριστικό "done" δέχεται μια συνάρτηση που θα τρέξει όταν ολοκληρωθεί η ασύγχρονη διαδικασία.

```
var deferred = Q.defer();
if (_myObj.isClear) {
    jsdom.env({
        file : _myObj.file,
        scripts : ["http://code.jquery.com/jquery.js"],
        done : function (err, window) {
            if (err) {
                logMe('createResObject.callback', 'err: '+err);
                deferred.reject(err);
            }
            var $ = '';
            if (window.$)
             $ = window.$ ;
            else if ( window.jQuery )
                $ = window.jQuery;
            var _head = $("head").html();
            var _body = $("body").html();
            $('script').last().remove();
            deferred.resolve({ head : _head, body: _body});
        }
    });
}
```

Εικόνα 52 - Δημιουργία τελικής σελίδας μέσω jsdom

Στην περίπτωση όμως που η αρχική σελίδα έχει χαρακτηριστεί ως ύποπτη ή ως κακόβουλη, τότε η εφαρμογή δημιουργεί μια νέα καθαρή ιστοσελίδα για να παρουσιάσει στον χρήστη, στην οποία αναφέρει τους λόγους που το Webninja χαρακτήρισε επικίνδυνη την σελίδα.

Webninja detected malicious/suspicious content

malicious: MSOfficeSnapshotViewer CVE-2008-2463 detected F0E42D60-368C-11D0-AD81-00A0C90DC8D9
malicious: MSIENestedSpan CVE-2008-4844 detected

Εικόνα 53 - Αποτέλεσμα σε περίπτωση κακόβουλης σελίδας

Συγκεκριμένα, η εφαρμογή αφού ελέγξει ότι έχει εκχυθεί σωστά το jQuery θα δημιουργήσει 1 αντικείμενο Javascript με 2 χαρακτηριστικά, head και body. Στο head θα αναθέσει τον κώδικα του head html αντικειμένου, ενώ στο body τον κώδικα του body html αντικειμένου.

Τέλος, το αντικείμενο αυτό επιστρέφεται στον χρήστη και ξανακάνοντας χρήση του shelljs διαγράφονται οτιδήποτε προσωρινά αρχεία έχουν δημιουργηθεί από την μέχρι τώρα διαδικασία.

```
    .then(function(html) {
        logMe('app.post', 'before response');

        res.json(html);
        shell.rm("-rf", shell.tempdir() + "/" + userId + ".tmp.txt");
        shell.rm("-rf", shell.tempdir() + "/" + userId + ".html");
        shell.rm("-rf", "~/webNinjaOutput/" + userId);
        shell.rm("-rf", path.join(__dirname, userId+".html"));
    })
    .done();
```

Εικόνα 54 - Επιστροφή αποτελέσματος στον χρήστη

Το server μέρος της εφαρμογής, είναι φτιαγμένο έτσι ώστε να μπορεί να εγκατασταθεί σε κάποιο κεντρικό μηχάνημα, server, το οποίο να δέχεται κλήσεις από τις διάφορες εγκαταστάσεις της επέκτασης είτε να εγκατασταθεί τοπικά στον προσωπικό υπολογιστή ή προσωπικό server του χρήστη. Πρακτικά λειτουργεί σαν ένας proxy server που φιλτράρει το περιεχόμενο της κάθε ιστοσελίδας την οποία επισκέπτεται ο χρήστης.

Για την εγκατάσταση του server της εφαρμογής, έχει δημιουργηθεί ένα δημόσιο Docker repository στο Docker Hub. Η εφαρμογή μπορεί να «κατέβει» στο μηχάνημα εκτελώντας την εντολή:

```
docker pull rousojohn/webninjaserver
```

Εικόνα 55 - Κατέβασμα του Webninja server ως docker container

Εναλλακτικά, δεν συνίσταται, μπορεί κάποιος να κατεβάσει την εφαρμογή από τον παρακάτω σύνδεσμο και να την εγκαταστήσει τοπικά σε κάποιο μηχάνημα.

https://github.com/rousojohn/masterDiploma.git

Προϋποθέσεις για αυτή την λύση είναι το Λειτουργικό Σύστημα να είναι κάποιο Linux distribution στο οποίο να τρέχει το jsunpack-n και το τελευταίο να είναι εγκατεστημένο στο home folder του χρήστη. Εγκατεστημένο, επίσης θα πρέπει να είναι το node.js.

Για την δημιουργία του Docker container, ακολουθήθηκε η παρακάτω διαδικασία. Όλη η διαδικασία της ανάπτυξης της εφαρμογής έγινε σε περιβάλλον Ubuntu 16.10. Σε αυτό εγκαταστάθηκε το Docker. Χρησιμοποιώντας την εντολή

```
docker pull ubuntu
```

Εικόνα 56 - Κατέβασμα ubuntu docker

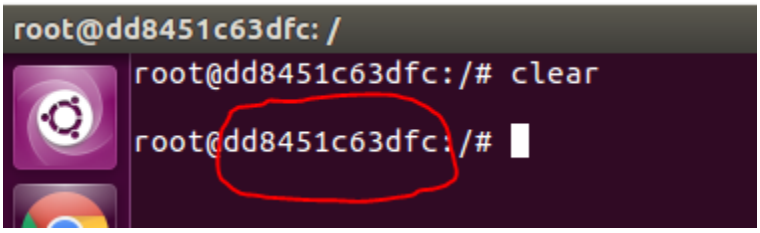
«κατεβάσαμε» τοπικά ένα container με την τελευταία έκδοση του Λειτουργικού Συστήματος Ubuntu από το Docker Hub. Σκοπός είναι να προ-εγκαταστήσουμε το jsunpack.

Με την εντολή

```
docker run -it ubuntu /bin/bash
```

Εικόνα 57 - Εναρξη και αλληλεπίδραση με το container

Τρέχουμε το container και αλληλοεπιδρούμε μαζί του μέσω γραμμής εντολών, όπως αν είχαμε συνδεθεί με SSH. Μόλις ενεργοποιηθεί το container και ανοίξει η σύνδεση, κρατάμε κάπου το ID του image. Το ID αυτό μπορούμε να το βρούμε δίπλα από το όνομα χρήστη που έχει κάνει είσοδο στο container και κατά 90% είναι ο root χρήστης. Το ID θα χρησιμοποιηθεί αργότερα για την αποθήκευση των αλλαγών στο container.



Εικόνα 58 - ImageID

Εκτελούμε τις απαραίτητες ενημερώσεις και κατεβάζουμε και εγκαθιστούμε το node.js όπως και το jsunpack (https://github.com/urule99/jsunpack-n.git) στο home folder του χρήστη, στην συγκεκριμένη περίπτωση στο «/root». Αφού ολοκληρώσουμε την εγκατάσταση και βεβαιωθούμε ότι το jsunpack τρέχει σωστά στο σύστημα σταματάμε το container και επιστρέφουμε στο τοπικό μας μηχάνημα. Από εκεί εκτελούμε την εντολή

```
docker commit <image_id> <username/container>
```

Εικόνα 59 - Αποθήκευση των αλλαγών στο image

στην οποία ως image_id βάζουμε το ID που κρατήσαμε προηγουμένως και στο username / container βάζουμε το όνομα χρήστη μας στο Docker Hub και ένα χαρακτηριστικό όνομα για το νέο container που δημιουργήσαμε. Στην συνέχεια δημιουργούμε το λεγόμενο Dockerfile. Το Dockerfile είναι ένα αρχείο που περιέχει όλες τις εντολές που θα χρησιμοποιούσε κάποιος για να παράγει ένα Docker image. Ουσιαστικά είναι κάτι σαν script ή πιο καλή παρομοίωση θα ήταν πως είναι ακριβώς ότι είναι και το αρχείο SPEC για την παραγωγή ενός RPM πακέτου. Το Dockerfile λοιπόν, στην περίπτωσή μας είναι κάπως έτσι:

```
FROM webninjaserver

RUN mkdir -p /root/webninjaserver/
WORKDIR /root/webninjaserver/

COPY package.json /root/webninjaserver/

RUN npm install

COPY nodeServer.js /root/webninjaserver/

EXPOSE 8081

CMD ["node", "nodeServer.js"]
```

Εικόνα 60 - Webninja Dockerfile

Πρακτικά, οι παραπάνω εντολές λένε στο Docker να χρησιμοποιήσει σαν αρχικό image το image με όνομα «webninjaserver». Έτσι το είχαμε ονομάσει πιο πριν με την εντολή commit. Αμέσως μετά δημιουργεί έναν φάκελο κάτω από τον αρχικό φάκελο του user (root) και πλοηγείται σε αυτόν. Εκεί αντιγράφει από τον τοπικό δίσκο στον φάκελο που βρίσκεται το αρχείο «package.json». Το αρχείο αυτό είναι ένα αρχείο που περιέχει metadata σχετικά με την εφαρμογή, όπως εξαρτήσεις και plugins που χρησιμοποιεί. Αφού αντιγραφτεί το αρχείο στο image, εκτελείται η εντολή npm install με την οποία εγκαθιστώνται στο image, όλες οι εξαρτήσεις και τα modules της εφαρμογής που αναφέρονται στο package.json. Εν συνεχεία, αντιγράφεται το script της εφαρμογής, ο κώδικας και επειδή πρόκειται για http Server που έχει δηλωθεί να δέχεται συνδέσεις στην πόρτα 8081, προωθούμε την πόρτα μέσω της εντολής EXPOSE.

Τέλος λέμε στο Docker όταν ξεκινά το image να εκτελεί την εντολή node nodeServer.js, δηλαδή την εφαρμογή μας.

9. Μελλοντικές Επεκτάσεις – Βελτιώσεις

Όπως κάθε εφαρμογή που δημιουργείται έτσι και το Webninja, δέχεται περεταίρω προσθήκες και βελτιώσεις.

Αρχικά καλό θα ήταν να βελτιωθεί ο χρόνος απόκρισης της εφαρμογής. Να γίνει πιο γρήγορη δηλαδή η ανάλυση και ο «καθαρισμός» των ιστοσελίδων που επισκέπτεται ο χρήστης. Αυτό θα μπορούσε να επιτευχθεί με δύο τρόπους:

1. Δημιουργία εργαλείου ανάλυσης javascript κώδικα
   Θα μπορούσε να αντικατασταθεί η χρήση του jsunpack από ένα εργαλείο ανάλυσης javascript κώδικα που θα ήταν μέρος του server του Webninja και όχι τρίτο εργαλείο στο οποίο θα εξαρτάται.

2. Επέκταση του jsunpack.
   Μια άλλη λύση θα ήταν να επεκταθεί το jsunpack έτσι ώστε να εκθέτει δημόσια κάποιο web service το οποίο θα καλεί η client (chrome extension) εφαρμογή. Το jsunpack θα μετατραπεί έτσι ώστε αντί να παράγει αρχεία με τον αυθεντικό και τον αποκρυπτογραφημένο κώδικα, να αναγνωρίζει τα κακόβουλα σημεία και να τα αφαιρεί απ'ευθείας από την είσοδο που δέχεται. Όλη η διαδικασία δηλαδή θα γίνεται σε ένα και μόνο βήμα χωρίς ενδιάμεσες εγγραφές και αναγνώσεις στον δίσκο του συστήματος, οι οποίες καθυστερούν πάρα πολύ την εφαρμογή.

Μια άλλη επέκταση θα ήταν να προστεθεί κάποια Βάση Δεδομένων (ίσως MongoDB λόγω ταχύτητας και γρήγορης αναζήτησης). Η ΒΔ αυτή θα μπορούσε να χρησιμοποιηθεί σαν cache μνήμη, αποθηκεύοντας hashes του αυθεντικού κώδικα και του URL της σελίδας καθώς και ο καθαρός κώδικας που αντιστοιχεί. Όταν θα γίνεται μια κλήση προς τον server θα ελέγχονται τα hashes και όπως σε μια cache μνήμη αν υπάρχει «hit» θα επιστρέφεται στον χρήστη η καθαρή σελίδα, ενώ σε περίπτωση «miss» θα υπολογίζονται όλα κανονικά και θα αποθηκεύονται στην ΒΔ για μελλοντική χρήση.

Το Webninja θα μπορούσε να επεκταθεί έτσι ώστε να υποστηρίζει πολλαπλούς περιηγητές διαδικτύου και όχι μόνο τον Google Chrome. Καλό θα ήταν να δημιουργηθεί επέκταση τόσο για τον Mozilla Firefox όσο και για τον Internet Explorer.

Επίσης μια πολύ ωραία πρόταση είναι να δημιουργηθεί κάποιο είδους reporting (αναφορών – στατιστικών). Για παράδειγμα, ένας χρήστης θα μπορούσε να κάνει «κλικ» στο κουμπί της επέκτασης Webninja και να του παρουσιάζεται κάποιο μενού με γραφήματα που να του υποδεικνύει ποιες ιστοσελίδες επισκέπτεται συνήθως, πόσες από αυτές βρέθηκαν ύποπτες κτλ.

Τέλος μια πιθανή προσθήκη θα ήταν η προσθήκη configuration από τον χρήστη, Να μπορεί για παράδειγμα να αποκλείσει το Webninja να τρέξει για κάποια ιστοσελίδα ή να τρέχει πάντοτε για κάποια άλλη άσχετα με το αν υπάρχει στην cache (whitelists και blacklists).

10. Πίνακας εικόνων

11. Βιβλιογραφία

(n.d.). Ανάκτηση 2016, από Hack In The Box: http://magazine.hackinthebox.org/

Bursztein, E. (2010, June 17). *Busting Frame Busting a Study of Clickjacking Vulnerabilities on Popular Sites.* Ανάκτηση 2016, από Elie: https://cdn.elie.net/publications/busting-frame-busting-a-study-of-clickjacking-vulnerabilities-on-popular-sites.pdf

Comodo. (2013). Help about Web Insector. Clifton, New Jersey, United States Of America: Comodo. Ανάκτηση από https://help.comodo.com/topic-208-1-490-5111-.html

Comodo. (2013, 11). Tis the Season for Web Inspector to Protect your Customers and Defend your Company's Online Reputation. Clifton, New Jersey, United States of America. Ανάκτηση από https://www.comodo.com/news/press_releases/2013/11/comodo-web-inspector-with-advanced-malware-detection-technologies.html

Comodo. (2016). Ανάκτηση από Web Inspector: https://www.webinspector.com/

Göbel, J. G., & Dewald, A. (2011). *Client-Honeypots: Exploring Malicious Websites.* (K. Mönch, Επιμ.) Munchen: R. Oldenbourg Verlag. Ανάκτηση από https://books.google.gr/books?id=3BhPZi0C2EUC&printsec=frontcover&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false

Google Inc. (n.d.). Developer Chrome. Mountain View, California, United States of America. Ανάκτηση 2016, από https://developer.chrome.com/home

jsunpack-n. (2010, May 19). Ανάκτηση 2016, από https://github.com/urule99/jsunpack-n

Nazario, J. (n.d.). Reverse Engineering Malicious Javascript. Ανάκτηση από https://cansecwest.com/csw07/csw07-nazario.pdf

Node.js Foundation. (2016). Ανάκτηση από Node.js: https://nodejs.org/en/

Node.js Foundation. (n.d.). *Express*. Ανάκτηση 2016, από http://expressjs.com/

npm, Inc. (2014). *npm*. Ανάκτηση 2016, από https://www.npmjs.com/

OWASP. (n.d.). Ανάκτηση 2016, από https://www.owasp.org/index.php/Main_Page

PC Advisor. (n.d.). London, England, United Kingdom. Ανάκτηση από http://www.pcadvisor.co.uk/

Sophos Ltd. (2011). Ανάκτηση από http://www.mgcwallace.com/wp-content/uploads/sophos-malicious-javascript-attacks-solutions-wpna.pdf

Symantec Corporation. (n.d.). *Symantec.* Ανάκτηση 2016, από https://support.symantec.com/en_US.html?redirect=false

Taute, S. (2012, June 22). *JsDetox*. (S. Taute, Επιμελητής) Ανάκτηση 2016, από http://www.relentless-coding.com/projects/jsdetox/

Zaharia, A. L. (2016, June 29). JavaScript Malware – a Growing Trend Explained for Everyday Users. Bucharest, Romania. Ανάκτηση από https://heimdalsecurity.com/blog/javascript-malware-explained/