Δίκτυα Υπολογιστών
Εργαστηριακή Άσκηση 12

Ονοματεπώνυμο: Κυριακόπουλος Γεώργιος – el18153
Ομάδα: 4
Όνομα PC/ΛΣ: George – Windows 10 Pro, 21H2
Ημερομηνία: 21/01/2021
Διεύθυνση IP: 192.168.1.10
Διεύθυνση MAC: 60-A4-B7-75-72-0F

Άσκηση 1:

1.1.
Ο εξυπηρετητής επιστρέφει τον κωδικό κατάστασης 401 – *Authorization Required* ως απόκριση στο αρχικό μήνυμα *HTTP* τύπου *GET*.

1.2.
Το δεύτερο μήνυμα *HTTP* τύπου *GET* περιέχει 2 νέα πεδία το *Cache-Control* και το *Authorization*.

1.3.
Οι τιμές των 2 αυτών πεδίων είναι *Cache-Control: max-age=0* και *Authorization: Basic ZWR1LWR5OnBhc3N3b3Jk* με υπο-πεδίο το *Credentials: edu-dy:password*.

1.4.
Το αποτέλεσμα της αποκωδικοποίησης είναι το *edu-dy:password*, που έχει συμπληρώσει και το Wireshark αυτόματα στο πεδίο *Credentials.*

1.5.
Ο βασικός αυτός μηχανισμός πιστοποίησης αυθεντικότητας που παρέχει το πρωτόκολλο *HTTP* δεν είναι ένας ασφαλής μηχανισμός, καθώς έχει έλλειψη εμπιστευτικότητας (confidentiality). Αυτό σημαίνει ότι οποιοσδήποτε ενδιάμεσος κόμβος έχει τη δυνατότητα να κατανοήσει μέσω μίας αποκωδικοποίησης τα μηνύματα που μεταφέρονται από τον αποστολέα στον παραλήπτη, όπως για παράδειγμα τα στοιχεία της επαλήθευσης της ταυτότητας του χρήστη.

Άσκηση 2:

2.1.
Το *SSH* χρησιμοποιεί το πρωτόκολλο μεταφοράς *TCP*.

2.2.
Οι θύρες που χρησιμοποιούνται είναι οι 65133 (στον υπολογιστή μου) και 22 (στον εξυπηρετητή).

2.3.
Η θύρα 22 αντιστοιχεί στο πρωτόκολλο εφαρμογής *SSH*.

2.4.
Χρησιμοποίησα το φίλτρο απεικόνισης *ssh*.

2.5.
Ο εξυπηρετητής χρησιμοποιεί την έκδοση *2.0* του πρωτοκόλλου *SSH* και την έκδοση *6.6.1_hpn13v11* του λογισμικού *OpenSSH*, με σχόλιο το λειτουργικό λογισμικό, *FreeBSD-20140420*.

2.6.
Ο πελάτης (ο υπολογιστής μου) χρησιμοποιεί την έκδοση 2.0 του πρωτοκόλλου *SSH* και την έκδοση *0.76* του λογισμικού *PuTTY*, ενώ δεν ακολουθεί κάποιο σχόλιο.

2.7.
Το πλήθος των αλγορίθμων ανταλλαγής κλειδιών (*kex*) που υποστηρίζει ο υπολογιστής μου (*client to server*) είναι 13 και οι δύο πρώτοι είναι οι *curve448-sha512, curve25519-sha256.*

2.8.
Το πλήθος των αλγορίθμων παραγωγής κλειδιών (*server host key*) που υποστηρίζει ο υπολογιστής (*client to server*) μου είναι 9 και οι δύο πρώτοι είναι οι *ssh-ed448, ssh-ed25519.*

2.9.
Το πλήθος των αλγορίθμων κρυπτογράφησης (*encryption*) που υποστηρίζει ο υπολογιστής μου (*client to server*) είναι 14 και οι δύο πρώτοι είναι οι *aes256-ctr, aes256-cbc*.

2.10.
Το πλήθος των αλγορίθμων πιστοποίησης αυθεντικότητας μηνυμάτων (*mac*) που υποστηρίζει ο υπολογιστής μου είναι 8 και οι δύο πρώτοι είναι οι *hmac-sha2-256, hmac-sha1*.

2.11.
Το πλήθος των αλγορίθμων συμπίεσης (*compression*) που υποστηρίζει ο υπολογιστής μου (*client to server*) είναι 3 και οι δύο πρώτοι είναι οι *none, zlib*.

2.12.
Ο αλγόριθμος ανταλλαγής κλειδιών που θα ακολουθήσουν τα δύο μέρη είναι ο *curve25519-sha256*, τον οποίο εμφανίζει και το Wireshark δίπλα από το πεδίο *Key Exchange*.

2.13.
Ουσιαστικά ψάχνουμε για τον πρώτο αλγόριθμο κρυπτογράφησης που υποστηρίζει ο πελάτης, ο οποίος υπάρχει και στη λίστα των υποστηριζόμενων από τον εξυπηρετητή. Αυτός είναι ο *aes256-ctr*.

2.14.
Ουσιαστικά ψάχνουμε για τον πρώτο αλγόριθμο πιστοποίησης αυθεντικότητας μηνυμάτων που υποστηρίζει ο πελάτης, ο οποίος υπάρχει και στη λίστα των υποστηριζόμενων από τον εξυπηρετητή. Αυτός είναι ο *hmac-sha2-256*.

2.15.
Ουσιαστικά ψάχνουμε για τον πρώτο αλγόριθμο συμπίεσης που υποστηρίζει ο πελάτης, ο οποίος υπάρχει και στη λίστα των υποστηριζόμενων από τον εξυπηρετητή. Αυτός είναι ο *none*.

2.16.
Το Wireshark εμφανίζει την παραπάνω επιλογή των αλγορίθμων κρυπτογράφησης, πιστοποίησης αυθεντικότητας μηνυμάτων και συμπίεσης δίπλα από το πεδίο *SSH Version 2*.

2.17.
Οι άλλοι τύποι μηνυμάτων *SSH* που καταγράφηκαν είναι οι *Elliptic Curve Diffie-Hellman Key Exchange Init, Elliptic Curve Diffie-Hellman Key Exchange Reply, New Keys, Encrypted packet*.

2.18.
Δεν μπορώ να διακρίνω σε ποια πακέτα μεταφέρεται η πληροφορία για την προτροπή *login* και *password* στην περίπτωση του *SSH*, αφού τα σχετικά πακέτα είναι κρυπτογραφημένα.

2.19.
Το πρωτόκολλο *SSH* είναι μία ασφαλής επιλογή σε σχέση με άλλα πρωτόκολλα ανταλλαγής δεδομένων, όπως το *HTTP*. Με τη χρήση των κλειδιών (δημοσίων και ιδιωτικών για κάθε χρήστη), την κρυπτογράφηση των μηνυμάτων και τη χρήση συμπίεσης καλύπτει τα θέματα της πιστοποίησης αυθεντικότητας (*authentication*), του ελέγχου πρόσβασης (*access control*), της εμπιστευτικότητας (*confidentiality*), της ακεραιότητας (*integrity*) και του απορρήτου (*privacy*).

Άσκηση 3:

3.1.
Χρησιμοποίησα το φίλτρο σύλληψης *host bbb2.cn.ntua.gr*.

3.2.
Χρησιμοποίησα το φίλτρο απεικόνισης *tcp.flags.syn == 1 and ip.src == 192.168.1.10*.

3.3.
Οι συνδέσεις γίνονται στις θύρες 80 και 443 του εξυπηρετητή *bbb2.cn.ntua.gr*.

3.4.
Η θύρα 80 αντιστοιχεί στο πρωτόκολλο εφαρμογής *HTTP* και η θύρα 443 στο πρωτόκολλο εφαρμογής *HTTPS*.

3.5.
Στην περίπτωση του *HTTP* έγιναν 6 συνδέσεις *TCP*, ενώ στην περίπτωση του *HTTPS* έγινε 1 σύνδεση *TCP*.

3.6.
Η θύρα πηγής για την περίπτωση του *HTTPS* είναι η 50751.

3.7.
Οι επικεφαλίδες Στρώματος Εγγράφων TLS έχουν όλες 3 κοινά πεδία τα *Content Type, Version* και *Length* με μήκος αντίστοιχα 1, 2 και 2 byte.

3.8.
Τα διαφορετικά πρωτόκολλα *TLS* του πεδίου *Content Type* (με τις αντίστοιχες τιμές τους) είναι τα εξής: *Handshake – 22*, *Change Cipher Spec – 20*, και *Application Data – 23*.

3.9.
Οι διαφορετικοί τύποι μηνυμάτων χειραψίας που παρατήρησα είναι οι εξής: *Client Hello, Server Hello, Certificate, Server Key Exchange, Server Hello Done, Client Key Exchange, Encrypted Handshake Message* και *New Session Ticket.*

3.10.
Έστειλε 1 μήνυμα, αφού έγινε μία μόνο σύνδεση *TCP* με *HTTPS*.

3.11.
Η μέγιστη έκδοση του *TSL* που υποστηρίζει ο πελάτης είναι η *TLS 1.2*.

3.12.
Ο τυχαίος αριθμός που περιέχει το πρώτο μήνυμα *Client Hello* είναι 32 byte. Τα 4 πρώτα byte είναι τα 2b, 40, 61 και 58, ενώ υποτίθεται ότι παριστάνουν την ώρα αποστολής του πακέτου σε *Unix time* (δηλαδή είναι ίσα με τα δευτερόλεπτα που έχουν περάσει από την *00:00:00 UTC, 1 January 1970*), ωστόσο αυτό το χαρακτηριστικό έχει πλέον αφαιρεθεί, όπως φαίνεται και στην

περίπτωση μας που δίνει λάθος ημερομηνία (Dec 29, 1992 16:31:52.000000000 GTB Standard Time)

3.13.
Το πλήθος των σουιτών κωδικών (*cipher suites*) που υποστηρίζει ο πελάτης είναι 16, ενώ οι δεκαεξαδικές τιμές των δύο πρώτων από αυτές είναι οι *0xcaca* και *0x1303*.

3.14.
Η έκδοση του πρωτοκόλλου TLS που θα χρησιμοποιηθεί είναι η TLS 1.2, ενώ το όνομα της σουίτας κωδικών κρυπτογράφησης που θα χρησιμοποιηθεί είναι *TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256* με δεκαεξαδική τιμή *0xc02f*.

3.15.
Ο τυχαίος αριθμός που περιέχει το μήνυμα *Server Hello* είναι 32 byte. Τα 4 πρώτα του byte είναι 49, 5d, 42 και a3.

3.16.
Δεν χρησιμοποιείται κάποια μέθοδος συμπίεσης, όπως φαίνεται από το πεδίο *Compression Method* με τιμή *null (0)*.

3.17.
Οι αλγόριθμοι που θα χρησιμοποιηθούν είναι οι *ECDH* (ανταλλαγής κλειδιών), *RSA* (πιστοποίησης ταυτότητας), *AES_128_GCM* (κρυπτογράφησης) και για συνάρτηση κατακερματισμού η *SHA256*.

3.18.
Το μήκος της εγγραφής *TLS* που μεταφέρει το πιστοποιητικό (*certificate*) του εξυπηρετητή είναι 4278 byte, σύμφωνα με το πεδίο *length* της επικεφαλίδας της.

3.19.
Μεταφέρονται 3 πιστοποιητικά με ονόματα *R3* της *Let's Encrypt*, *ISRG Root X1* της *Internet Security Research Group* και *DST Root CA X3* της *Digital Signature Trust Co*.

3.20.
Χρειάστηκαν 4 πλαίσια Ethernet ώστε να μεταφερθεί η παραπάνω εγγραφή *TLS*.

3.21.
Ο εξυπηρετητής στέλνει ένα δημόσιο κλειδί μήκους 32 byte, όπως και ο πελάτης. Τα πρώτα 5 γράμματα των κλειδιών αντίστοιχα είναι *cc1ad* και *cf658*.

3.22.
Το μήκος της εγγραφής *TLS* που μεταφέρει στον εξυπηρετητή την υπόδειξη ότι η επικοινωνία από εδώ και πέρα θα είναι κρυπτογραφημένη (*ChangeCipherSpec*) είναι 6 byte.

3.23.
Το μήκος της εγγραφής *TLS* που μεταφέρει στον εξυπηρετητή το αποτέλεσμα της συνάρτησης

κατακερματισμού επί των προηγούμενων μηνυμάτων της χειραψίας (*EncryptedHandshakeMessage*) είναι 45 byte.

3.24.
Ναι, υπήρχαν εγγραφές *TLS* με την υπόδειξη *ChangeCipherSpec* και το αποτέλεσμα της συνάρτησης κατακερματισμού *EncryptedHandshakeMessage* από την πλευρά του εξυπηρετητή.

3.25.
Όχι, δεν υπήρχαν εγγραφές *TLS* του πρωτοκόλλου *Alert*.

3.26.
Δεν υπήρχαν.

3.27.
Υπάρχει αποτέλεσμα στην περίπτωση του *HTTP*, ενώ για το *HTTPS* όχι. Ο λόγος είναι, φυσικά, ότι το πρωτόκολλο *HTTPS* χρησιμοποιεί κρυπτογράφηση, οπότε δεν θα μπορώ να βρω αυτούσια τη φράση μέσα στα μηνύματα που ανταλλάχθηκαν, καθώς έχει κρυπτογραφηθεί.

3.28.
Το πρωτόκολλο *HTTPS* είναι αρκετά πιο ασφαλές από το *HTTP*, λόγω της ενσωμάτωσης σε αυτό ενός επιπλέον πρωτοκόλλου για την ασφαλή μετάδοση των δεδομένων (παλιότερα το *SSL*, πλέον το *TLS*). Με τη χρήση των πιστοποιητικών, την κρυπτογράφηση των μηνυμάτων και τη χρήση των hash functions καλύπτει τα θέματα της πιστοποίησης αυθεντικότητας (*authentication*), της εμπιστευτικότητας (*confidentiality*), της ακεραιότητας (*integrity*) και του απορρήτου (*privacy*). Σε αντίθετη περίπτωση, το πρωτόκολλο *HTTP* δεν παρέχει τίποτα από τα παραπάνω.