Όνοματεπώνυμο: Πυλιώτης Αθανάσιος		**Ομάδα: 3**
Όνομα PC/ΛΣ: DESKTOP-5DLG3IF	**Ημερομηνία:**	**26/12/22**
Διεύθυνση IP: 147.102.131.53	**Διεύθυνση MAC:**	**98:54:1b:bd:69:97**

Εργαστηριακή Άσκηση 12
Ασφάλεια

Απαντήστε στα ερωτήματα στον χώρο που σας δίνεται παρακάτω και στην πίσω σελίδα εάν δεν επαρκεί. Το φυλλάδιο αυτό θα παραδοθεί στον επιβλέποντα.

1

1.1 **Status code:** 401 Authorization Required\r\n

1.2 **WWW-Authenticate:** *Basic* realm="Edu-DY TEST"\r\n χρησιμοποιεί Basic στο WWW-Authenticate

1.3 **Authorization:** Basic ZWR1LWR5OnBhc3N3b3Jk\r\n και μετά

Credentials: edu-dy:password

Στέλνει authorization σε Basic όπως ζητήθηκε και μετά τα credentials

1.4

0000 41 75 74 68 6f 72 69 7a 61 74 69 6f 6e 3a 20 42 Authorization: B

0010 61 73 69 63 20 5a 57 52 31 4c 57 52 35 4f 6e 42 asic ZWR1LWR5OnB

0020 68 63 33 4e 33 62 33 4a 6b 0d 0a hc3N3b3Jk..

Δεν γράφει τα credentials αλλά μόνο το Basic (τύπο κωδικοποίησης) και τον κωδικό του.

1.5 ZWR1LWR5OnBhc3N3b3Jk → (decode) edu-dy:password

1.6 Ότι είναι σχεδόν ανύπαρκτη. Χρησιμοποιεί μια πολύ απλή κωδικοποίηση την οποία μπορεί ο οποιοσδήποτε να σπάσει αν προσπαθήσει λίγο. Προφανώς για αυτό χρειάστηκε αναβάθμιση της ασφάλειας στο διαδίκτυο και άλλαξε το πρωτόκολλο.

2

2.1 TCP

2.2 Transmission Control Protocol, Src Port: **60840**, Dst Port: **22**

2.3 Η θύρα 22 (Η άλλη θύρα είναι στη περιοχή ιδιωτικών θυρών)

2.4 ssh

2.5 Protocol: SSH-2.0-OpenSSH_6.6.1_hpn13v11 FreeBSD-20140420

Ο εξυπηρετητής χρησιμοποιεί την έκδοση 2.0 με software OpenSSH στην έκδοση 6.6.1 hpn13v11

το FreeBSD-20140420 που δείχνει το OS

Δεν υπάρχει κάτι άλλο

2.6 Protocol: SSH-2.0-OpenSSH_8.9p1 Ubuntu-3

Ο εξυπηρετητής χρησιμοποιεί την έκδοση 2.0 με software OpenSSH στην έκδοση 8.9p1

Το OS που χρησιμοποιεί ο πελάτης είναι το Ubuntu-3

Δεν υπάρχει κάτι άλλο

2.7 Υπάρχουν **11** αλγόριθμοι και οι δύο πρώτοι είναι: `curve25519-sha256,curve25519-sha256@libssh.org`

2.8 Υπάρχουν **15** αλγόριθμοι και οι δύο πρώτοι είναι: `ssh-ed25519-cert-v01@openssh.com,ecdsa-sha2-nistp256-cert-v01@openssh.com`

2.9 6 αλγόριθμοι και οι 2 πρώτοι είναι: `chacha20-poly1305@openssh.com,aes128-ctr`

2.10 10 αλγόριθμοι και οι 2 πρώτοι είναι: `umac-64-etm@openssh.com,umac-128-etm@openssh.com`

2.11 `none,zlib@openssh.com,zlib`

2.12 **Key Exchange** (method:curve25519-sha256@libssh.org) το γράφει στην αρχή αυτόν που θα χρησιμοποιηθεί εν τέλει. Το εμφανίζει το Wireshark και είναι στους 2 πρώτους και στα δύο (πρώτος στο αντίστοιχο μήνυμα του εξυπηρετητή και δεύτερος στου πελάτη).

2.13 `chacha20-poly1305@openssh.com` θα πρέπει να είναι καθώς είναι ο πρώτος της λίστας του πελάτη και υπάρχει στην λίστα του εξυπηρετητή.

`Client: encryption_algorithms_client_to_server string: chacha20-poly1305@openssh.com,aes128-ctr,aes192-ctr,aes256-ctr,aes128-gcm@openssh.com,aes256-gcm@openssh.com`

`Server: encryption_algorithms_client_to_server string: aes128-ctr,aes192-ctr,aes256-ctr,arcfour256,arcfour128,chacha20-poly1305@openssh.com,aes128-cbc,3des-cbc,blowfish-cbc,cast128-cbc,aes192-cbc,aes256-cbc,arcfour,rijndael-cbc@lysator.liu.se`

Πήρε το πρώτο αλγόριθμο που ζήτησε ο πελάτης, αφού και ο εξυπηρετητής των αποδέχτηκε σαν δυνατό αλγόριθμο. Αν δεν τον είχε ο εξυπηρετητής σαν υποστηριζόμενο, θα προσπαθούσε να δει αν ο δεύτερος αλγόριθμος του πελάτη είναι υποστηριζόμενος κοκ.

2.14 `umac-64-etm@openssh.com` πρώτος και στον πελάτη και στον εξυπηρετητή

2.15 none, άρα κανείς δεν θα χρησιμοποιηθεί.

2.16 Όχι δεν φαίνεται να τους εμφανίζει

2.17 Καταγράψαμε το μήνυμα «Elliptic Curve Diffie-Hellman Key Exhange Init» και «Elliptic Curve Diffie-Hellman Key Exhange Reply, New Keys». Και μετά ο πελάτης απάντησε με «New Keys».

2.18 Όχι δεν μπορούμε. Τα υπόλοιπα πακέτα είναι κρυπτογραφημένα μετά τη συμφωνία των δύο μεριών και για αυτό όλα τα πακέτα μας φαίνονται ίδια.

2.19 Σε σύγκριση με ό,τι άλλο έχουμε παρατηρήσει είναι πάρα πολύ ασφαλές. Αρχικά χρησιμοποιεί πληθώρα αλγορίθμων για κρυπτογράφηση, για mac, για συμπίεση και για πολλά άλλα που το κάνουν πολύ περίπλοκο. Επίσης, δίνει πάρα πολλές επιλογές και ο πελάτης έχει την ελευθερία να επιλέξει πρώτος τι θέλει και να δει τι κρυπτογράφηση μπορεί να έχει ανάλογα. Το γεγονός ότι ούτε εμείς δεν μπορούμε να δούμε τα πακέτα είναι καθησυχαστικό καθώς δείχνει πως όντως κανένας δεν μπορεί αρκετά εύκολα, αν δεν μαντέψει την κρυπτογράφηση, να καταγράψει τα δεδομένα. Επίσης επιβεβαιώνει την αυθεντικότητα με το public key που είναι αρκετά ευχάριστο.

3

3.1 Capture Filter: host 147.102.40.19

3.2 Display Filter: tcp.flags.syn == 1 and tcp.flags.ack == 0

3.3 Στην αρχή βλέπουμε την **80** και μετά η ιδιωτική θύρα αυξάνεται κατά ένα με κάθε νέα τριπλή χειραψία. Για το HTTPS η θύρα σύνδεσης είναι η **443** όπως γνωρίζουμε ήδη.

3.4 Η θύρα 80 στο HTTP και η θύρα 443 στο HTTPS

3.5 6 συνδέσεις στη περίπτωση του HTTP και 2 συνδέσεις στη περίπτωση του HTTPS

3.6 Θύρες πηγής: 60312 και 60313

3.7 Content Type (1 byte), Version(2 bytes), Length(2 bytes)

3.8 Handshake(22), Application Data(23), Change Cipher Spec(20),

3.9 TLS 1.0 (0x0301), TLS 1.2 (0x0303)

3.10 Client Hello(1), Server Hello(2), Certificate(11), Client Key Exchange(16), Server Hello Done(14), New Session Ticket(4),

3.11 2, όσες και οι τριπλές χειραψίες που έγιναν. Πρακτικά κάθε φορά που έκανε σύνδεση έστειλε και Client Hello.

3.12 Version: TLS 1.0 (0x0301). Ναι γιατί είχα παρατηρήσει πως υπήρχαν μερικά που χρησιμοποιούσαν διαφορετική έκδοση και το κατέγραψα. Τα περισσότερα πακέτα ωστόσο χρησιμοποιούν την έκδοση 1.2 .

3.13 Supported Version: TLS 1.2 (0x0303) και Supported Version: TLS 1.3 (0x0304) άρα δέχεται την έκδοση 1.3. Αυτές οι δύο δηλώνονται.

3.14 Extension: application_layer_protocol_negotiation (len=14)

Type: application_layer_protocol_negotiation (16)

Length: 14

ALPN Extension Length: 12

ALPN Protocol

ALPN string length: 2

ALPN Next Protocol: h2

ALPN string length: 8

ALPN Next Protocol: http/1.1

Τα πρωτόκολλά που χρησιμοποιούνται είναι το h2 και το http/1.1

3.15 32 bytes μήκος και είναι:
74d26581b4203ff5c251e9952989141b245f9fabb4c75bd936858a0c31ec91f3

Τα πρώτα 4 bytes είναι 74 d2 65 81 παριστάνει μια ημερομηνία, συγκεκριμένα Feb 9, 2032 15:20:01.000000000 GTB Standard Time, άρα 10 χρόνια από τώρα. Αποτελεί ένα τυχαίο POSIX timestamp.

3.16 16 cipher suites στο πλήθος και οι πρώτοι δύο είναι: 0x1301 και 0x1302

3.17 Version: TLS 1.2 (0x0303)

3.18 32 bytes και είναι Random:
1a4ecca66c3f62a5c72c62e24eab306b726c8a4fe98118bb54bcd54a1b661b32

Τα 4 πρώτα είναι 1a 4e cc a6 και αποτελεί ημερομηνία, την GMT Unix Time: Dec 27, 1983 13:16:54.000000000 GTB Standard Time. Αποτελεί κι αυτή ένα POSIX Timestamp που χρησιμοποιείται και πλέον είναι τυχαίο ως φαίνεται στο σύστημα, λογικά για περισσότερη ασφάλεια. Παλιά ήταν current timestamp.

3.19 Compression Method: null (0), άρα δεν χρησιμοποιείται από κανένα, έχει την ίδια επικεφαλίδα με τη τιμή 0 ίδια και στα 4 πακέτα.

3.20 Θα χρησιμοποιηθεί Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (**0xc02f**) Ανταλλαγή κλειδιών (KEX): ECDHE, Πιστοποίηση Ταυτότητας (Authentication): RSA_WITH_AES_128 , Κρυπτογράφησης (Cryptography): GCM, Συνάρτηση Κατακερματισμού (Hashing Function): SHA256

3.21 Certificates Length: 4269 (συνολικό μήκος certificates), όλο το Handshake πρωτόκολλο είναι 4276 bytes και όλο το TLSv1.2 Record Layer είναι 4281 bytes.

3.22 3 certificates with lengths 1574 bytes, 1306 bytes and 1380 bytes

3.23 [5 Reassembled TCP Segments (4281 bytes): #434(1276), #435(1355), #437(1355), #438(31), #440(264)] άρα 5 πλαίσια Ethernet

3.24 Το μήκος του δημοσίου κλειδιού είναι 32 bytes kai 078e0 για τον client, Το μήκος του δημοσίου κλειδιού είναι 32 bytes kai c0c77 για τον εξυπηρετητή

3.25 Το μήκος της εγγραφής είναι 6 bytes από τα 93 bytes του TLS μηνύματος (γιατί στέλνεται και Client Key Exchange και άλλο encrypted Handshake). Και το μήκος του μηνύματος είναι 1 Byte.

3.26 Το μήκος του μηνύματος είναι 40 bytes. (και προφανώς είναι encrypted)

3.27 ναι, 2 από αυτά κι όλας, που προηγούνται με New Session Ticket.

3.28 Hyper Text Transfer Protocol 2

3.29 Όχι δεν βρήκαμε

3.30 Μετά από μελέτη μάθαμε πως πρόκειται για TLS Notification που στέλνεται για να μας ειδοποιήσει πως το session μεταξύ των 2 συσκευών σταματάει και δεν υπάρχουν άλλα δεδομένα να σταλούν. Αυτό που μπορεί να συμβεί είναι να λάβουμε κρυπτογραφημένα μηνύματα μετά που έχει σταματήσει η σύνδεση και να μην καταλαβαίνουμε τι μας μεταφέρουν για αυτό το λόγο.

3.31 Αναζητούμε “core features”. Καταφέρνουμε να το βρούμε μόνο στην HTTP μορφή και όχι στην HTTS. Αυτό συμβαίνει λόγω της παραπάνω ασφάλειας που προσφέρει το πρωτόκολλο HTTPS και την κρυπτογραφίας που κάνει στα δεδομένα που έρχονται στον υπολογιστή μας.

3.32 Αρχικά παρέχει ασφάλεια στο σύστημα, σε αντίθεση με το HTTP. Κρυπτογραφεί τα δεδομένα με πολλούς και διαφορετικούς τρόπους ώστε η υποκλοπή τους να είναι ακόμα πιο απαιτητική και να μεταφέρονται ακριβώς όπως στάλθηκαν. Παρατηρούμε πως οι σελίδες φορτώνουν ακριβώς με τον ίδιο τρόπο. Απλά κανένας εξωτερικός δεν μπορεί εύκολα να πάρει τα δεδομένα μας, όπως καταλαβαίνουμε από το γεγονός ότι κι εμείς βλέπουμε μόνο application data από την εφαρμογή της κρυπτογραφίας και μετά. Φαίνεται σαν ένα εξαιρετικό πρωτόκολλο για την ασφάλεια που προσφέρει και την υπηρεσία που παρέχει.