12η Εργαστηριακή Άσκηση στα “Δίκτυα Υπολογιστών”
Κυριάκος Τσαρτσαράκος
AM: 03118054
IPv4 Address: 192.168.1.9
MAC Address: 00:e9:3a:0b:0b:f3

1. Πιστοποίηση αυθεντικότητας στο πρωτόκολλο HTTP

1.1) Το μήνυμα έχει status code το 401 και η φράση είναι Authorization Required.

1.2) Το επιπλέον πεδίο που περιλαμβάνει η επικεφαλίδα HTTP του δεύτερου μηνύματος είναι το πεδίο Authorization.

1.3) Το περιεχόμενο του παραπάνω πεδίου, όπως εμφανίζεται σε μορφή ASCII είναι το εξής:
Authorization: Basic ZWR1LWR50nBhc3N3b3Jk

1.4) Το αποτέλεσμα της αποκωδικοποίησης που προκύπτει είναι το εξής:
edu-dy:password

1.5) Συμπεραίνω ότι ο μηχανισμός ανταλλαγής των credentials που χρησιμοποιείται στο HTTP Base64 είναι αρκετά αδύναμος, καθώς έχει έλλειψη εμπιστευτικότητας (confidentiality), επομένως, οποιοσδήποτε ενδιάμεσος κόμβος μπορεί να μάθει τα στοιχεία ταυτοποίησης του χρήστη.

2. Υπηρεσία SSH - Secure SHell

2.1) Χρησιμοποιεί το πρωτόκολλο μεταφοράς TCP.

2.2) Οι θύρες που χρησιμοποιούνται είναι οι εξής:
- port 37342 (θύρα του υπολογιστή μου)
- port 22 (θύρα του σερβερ)

2.3) Αντιστοιχεί η θύρα 22.

2.4) Η σύνταξη του φίλτρου απεικόνισης που χρησιμοποίησα είναι η εξής: ssh

2.5)
- Η έκδοση πρωτοκόλλου του εξυπηρετητή είναι: SSH-2.0
- Η έκδοση του λογισμικού είναι: OpenSSH_6.6.1_hpn13v11
- Το σχόλιο είναι: FreeBSD-20140420

2.6)
- Η έκδοση πρωτοκόλλου του εξυπηρετητή είναι: SSH-2.0
- Η έκδοση του λογισμικού είναι: OpenSSH_8.2p1
- Το σχόλιο είναι: Ubuntu-4ubuntu0.3

2.7) Το πλήθος τους είναι 10, και οι δύο πρώτοι αλγόριθμοι είναι οι εξής:
- curve25519-sha256

- curve25519-sha256@libssh.org

2.8) Το πλήθος τους είναι 13 και οι δύο πρώτοι αλγόριθμοι είναι οι εξής:
- ecdsa-sha2-nistp256-cert-v01@openssh.com
- ecdsa-sha2-nistp384-cert-v01@openssh.com

2.9) Το πλήθος τους είναι 6 και οι δύο πρώτοι αλγόριθμοι είναι οι εξής:
- chacha20-poly1305@openssh.com
- aes128-ctr

2.10) Το πλήθος τους είναι ίσο με 10 και οι δύο πρώτοι αλγόριθμοι είναι οι εξής:
- umac-64-etm@openssh.com
- umac-128-etm@openssh.com

2.11) Το πλήθος τους είναι ίσο με 3 και οι δύο πρώτοι αλγόριθμοι είναι οι εξής:
- none
- zlib@openssh.com

2.12) Είναι ο ecdh-sha2-nistp256, και τον εμφανίζει το Wireshark σε παρένθεση δίπλα στο πεδίο Key Exchange. (Key Exchange (method:ecdh-sha2-nistp256)).

2.13) Είναι ο αλγόριθμος aes128-ctr.

2.14) Είναι ο αλγόριθμος umac-64@openssh.com.

2.15) Είναι ο αλγόριθμος none.

2.16) Ναι, εμφανίζει σε παρένθεση δίπλα στο πεδίο SSH Version 2. (SSH Version 2 (encryption:aes128-ctr mac:umac-64@openssh.com compression:none)).

2.17) Κατέγραψα τους εξής υπόλοιπους τύπους μηνυμάτων: ”Elliptic Curve Diffie-Hellman Key Exchange Init”, ”Elliptic Curve Diffie-Hellman Key Exchange Reply”, ”New Keys”, ”Encrypted Packet”.

2.18) Όχι, καθώς είναι κρυπτογραφημένα.

2.19)
- Authentication: Μέσω της χρήσης public-private keys.
- Access control: Μέσω της χρήσης public-private keys.
- Confidentiality: Μέσω της κρυπτογράφησης των μηνυμάτων.
- Integrity: Με την συμπίεση compress και Mac.
- Privacy: Με την δημιουργία κοινού μυστικού κλειδιού

3. Υπηρεσία HTTPS

3.1) Η σύνταξη του φίλτρου σύλληψης που χρησιμοποίησα είναι η εξής:
host bbb2.cn.ntua.gr

3.2) Η σύνταξη του φίλτρου απεικόνισης που χρησιμοποίησα είναι η εξής:
tcp.flags.syn == 1

3.3) Για την http έγινε στην θύρα 80, ενώ για την https έγινε στην θύρα 443.

3.4) Στο πρωτόκολλο εφαρμογής HTTP αντιστοιχεί η θύρα 80, ενώ στο πρωτόκολλο εφαρμογής HTTPS αντιστοιχεί η θύρα 443.

3.5) Για http έγιναν 6 συνδέσεις, ενώ για https έγινε μία σύνδεση.

3.6) Η θύρα πηγής (source port) για τις συνδέσεις TCP στην περίπτωση HTTPS είναι η θύρα 34962.

3.7) Είναι τα εξής πεδία:
- Content Type: 1 byte
- Version: 2 bytes
- Length: 2 bytes

3.8) Οι διάφορες τιμές που προκύπτουν για το πεδίο Content Type είναι οι εξής:
- Handshake (22)
- Application Date (23)
- Alert (21)

3.9) Οι διαφορετικοί τύποι μηνυμάτων που παρατηρήθηκαν είναι οι εξής:
- Client Hello
- Server Hello
- Certificate
- Server Key Exchange
- Server Hello Done
- Client Key Exchange
- Change Cipher Spec
- Encrypted Handshake Message
- New Session Ticket

3.10) Έστειλε ένα μήνυμα Client Hello, το οποίο ουσιαστικά αντιστοιχεί σε μία σύνδεση TCP.

3.11) Είναι η TLSv1.2

3.12) Το μήκος του τυχαίου αριθμού είναι ίσο με 32 bytes, τα πρώτα τέσσερα bytes είναι τα εξής:
- 1ο byte: dc
- 2ο byte: 75
- 3ο byte: 7b
- 4ο byte: bf

Υπο κανονικές συνθήκες οι παραπάνω αριθμοί παριστάνουν την χρονική στιγμή της αποστολής του πακέτου, όπου πλέον δεν γίνεται, καθώς η χρονική στιγμή είναι απλά τυχαία.

3.13) Το πλήθος των cipher suites είναι ίσο με 17, και οι δεκαεξαδικές τιμές των πρώτων δύο είναι οι εξής:

- 1ο cipher suite: 0x1301
- 2ο cipher suite: 0x1303

3.14) Η έκδοση που θα χρησιμοποιηθεί είναι η TLSv1.2, ενώ η σουίτα κωδικών κρυπτογράφησης που επιλέχθηκε είναι η εξής:

- Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f)

3.15) Το μήκος του τυχαίου αριθμού είναι ίσο με 32 bytes, ενώ τα πρώτα τέσσερα bytes του τυχαίου αριθμού είναι τα εξής:

- 1ο byte: 58
- 2ο byte: b2
- 3ο byte: 19
- 4ο byte: fe

3.16) Όχι, δεν χρησιμοποιείται (Compression Method: null (0)).

3.17) Είναι οι εξής:

- Αλγόριθμος ανταλλαγής κλειδιών: ECDHE
- Αλγόριθμος πιστοποίησης ταυτότητας: RSA
- Αλγόριθμος κρυπτογράφησης: AES_128_GCM
- Συνάρτηση κατακερματισμού: SHA_256

3.18) Το μήκος της, σύμφωνα με το πεδίο Length, είναι ίσο με 4278 bytes.

3.19) Μεταφέρονται τρία πιστοποιητικά, τα οποία είναι τα εξής:

- bbb2.cn.ntua.gr
- R3
- ISRG Root X1

3.20) Χρειάστηκαν τέσσερα (4) πλαίσια Ethernet.

3.21) Αναλυτικά προκύπτουν:

- Πελάτης:
  - ☐ Μήκος κλειδιού: 32 bytes
  - ☐ Πέντε (5) πρώτα γράμματα: afd66
- Εξυπηρετητής:
  - ☐ Μήκος κλειδιού: 32 bytes
  - ☐ Πέντε (5) πρώτα γράμματα: 631a5

3.22) Έχει μήκος ίσο με 6 bytes.

3.23) Έχει μήκος ίσο με 45 bytes.

3.24) Ναι, παρατήρησα.

3.25) Ναι, παρατηρήθηκαν και στάλθηκαν μόνο από την πλευρά του πελάτη.

3.26) Υπάρχουν, καθώς ακολουθεί τερματισμός της σύνδεσης με την συγκεκριμένη θύρα.

3.27) Η αναζήτηση βρίσκει αποτέλεσμα μόνο για το πρωτόκολλο http και όχι για το https.

3.28) Στο πρωτόκολλο HTTPS έχουμε:

- Πιστοποίηση της αυθεντικότητας: Με την χρήση των certificates.
- Εμπιστευτικότητα: Με την κρυπτογράφηση των δεδομένων.
- Ακεραιότητα των δεδομένων: Με την χρήση των hash functions. Αντιθέτως, στο HTTP δεν έχουμε τίποτα από τα παραπάνω.