Δίκτυα Υπολογιστών

Εργαστηριακή Άσκηση 12
Ασφάλεια

Ονοματεπώνυμο: Νικόλαος Παγώνας, el18175	**Ομάδα:** 4 (Τρίτη εξ' αποστάσεως)
Όνομα PC/ΛΣ: nick-ubuntu/Ubuntu 20.04.3 LTS	**Ημερομηνία:** Τρίτη 18/01/2022
Διεύθυνση IP: `192.168.1.15`	**Διεύθυνση MAC:** `3c:2c:30:e1:1c:55`

1. Πιστοποίηση αυθεντικότητας στο πρωτόκολλο HTTP

1.1

Ο αριθμητικός κωδικός κατάστασης είναι 401 και η φράση είναι "Authorization Required".

1.2

Η επικεφαλίδα του δεύτερου μηνύματος περιέχει το επιπλέον πεδίο `Authorization`.

1.3

Είναι Authorization: `Basic ZWR1LWR5OnBhc3N3b3Jk`.

1.4

Το αποτέλεσμα της αποκωδικοποίησης είναι `edu-dy:password`.

1.5

Ο μηχανισμός πιστοποίησης αυθεντικότητας που παρέχει το HTTP δεν παρέχει εμπιστευτικότητα, αφού δεν έχουμε κανενός είδους κρυπτογράφηση.

2. Υπηρεσία SSH - Secure SHell

Σημείωση: Επειδή για κάποιον λόγο το ssh από το μηχάνημά μας αποτύγχανε, συνδεθήκαμε στο VPN του ΕΜΠ για την εκτέλεση του ερωτήματος 2.

2.1

Το SSH χρησιμοποιεί TCP.

2.2

Χρησιμοποιούνται οι θύρες 22 και 46636.

2.3

Η θύρα που αντιστοιχεί στο SSH είναι η 22.

2.4

Η σύνταξη του φίλτρου είναι `ssh`.

2.5

Η έκδοση πρωτοκόλλου SSH που χρησιμοποιεί ο εξυπηρετητής είναι η 2.0, ενώ η έκδοση του λογισμικού που χρησιμοποιεί είναι `OpenSSH_6.6.1_hpn13v11`. Επίσης περιλαμβάνεται το σχόλιο `FreeBSD-20140420`.

2.6

Η έκδοση πρωτοκόλλου SSH που χρησιμοποιεί ο πελάτης είναι η 2.0, ενώ η έκδοση του λογισμικού που χρησιμοποιεί είναι `OpenSSH_8.2p1`. Επίσης περιλαμβάνεται το σχόλιο `Ubuntu-4ubuntu0.3`

2.7

Υπάρχουν 10 αλγόριθμοι στη λίστα. Οι δύο πρώτοι από τους αλγορίθμους αυτής της λίστας είναι οι `curve25519-sha256` και `curve25519-sha256@libssh.org`.

2.8

Υπάρχουν 18 αλγόριθμοι στη λίστα. Οι δύο πρώτοι από τους αλγορίθμους αυτής της λίστας είναι οι `ecdsa-sha2-nistp256-cert-v01@openssh.com` και `ecdsa-sha2-nistp384-cert-v01@openssh.com`

2.9

Οι δύο πρώτοι αλγόριθμοι είναι οι `chacha20-poly1305@openssh.com` και `aes128-ctr`.

2.10

Οι δύο πρώτοι αλγόριθμοι είναι οι `umac-64-etm@openssh.com` και `umac-128-etm@openssh.com`.

2.11

Οι δύο πρώτοι αλγόριθμοι είναι οι `none` και `zlib@openssh.com`.

2.12

Ο αλγόριθμος που θα χρησιμοποιηθεί είναι ο `curve25519-sha256@libssh.org`, και αυτό φαίνεται επίσης στο πεδίο "Key Exchange", σε παρένθεση (`method: curve25519-sha256@libssh.org`).

2.13

Ο αλγόριθμος που θα χρησιμοποιηθεί είναι ο `chacha20-poly1305@openssh.com`.

2.14

Ο αλγόριθμος που θα χρησιμοποιηθεί είναι ο `umac-64-etm@openssh.com`.

2.15

Ο αλγόριθμος που θα χρησιμοποιηθεί είναι ο `none`.

2.16

Το Wireshark εμφανίζει τους επιλεχθέντες αλγορίθμους σε παρένθεση δίπλα από το πεδίο SSH Version 2.

2.17

Καταγράψαμε τους τύπους:

- Elliptic Curve Diffie-Hellman Key Exchange Init
- Elliptic Curve Diffie-Hellman Key Exchange Reply
- New Keys
- Encrypted packet

2.18

Δεν μπορώ, γιατί τα πακέτα είναι κρυπτογραφημένα.

2.19

Το SSH είναι πιο ασφαλές από πρωτόκολλα όπως το HTTP ή το TELNET. Συγκεκριμένα, όσον αφορά την πιστοποίηση αυθεντικότητας, αυτή γίνεται με την χρήση public-private keys, η εμπιστευτικότητα επιτυγχάνεται με την κρυπτογράφηση των μηνυμάτων, ενώ η ακεραιότητα εξασφαλίζεται με το MAC.

3. Υπηρεσία HTTPS

3.1

Η σύνταξη του φίλτρου σύλληψης που χρησιμοποιήσαμε είναι `host bbb2.cn.ntua.gr`.

3.2

Η σύνταξη του φίλτρου είναι `tcp.flags.syn == 1 && tcp.flags.ack == 0`.

3.3

Οι συνδέσεις έγιναν στις θύρες 80 και 443.

3.4

Η θύρα 80 αντιστοιχεί στο HTTP και η θύρα 443 στο HTTPS.

3.5

Στην περίπτωση HTTP έγιναν 6 συνδέσεις, ενώ στην περίπτωση HTTPS έγινε 1 σύνδεση.

3.6

Στην περίπτωση HTTPS έχουμε ως θύρα πηγής την 51698.

3.7

Τα τρία κοινά πεδία είναι:

- Content Type: 1 byte
- Version: 2 bytes
- Length: 2 bytes

3.8

Οι διαφορετικές τιμές που καταγράφηκαν είναι:

- Change Cipher Spec - 20
- Alert - 21
- Handshake - 22
- Application Data - 23

3.9

Οι διαφορετικοί τύποι μηνυμάτων χειραψίας είναι:

- Client Hello - 1
- Server Hello - 2
- New Session Ticket - 4
- Certificate - 11
- Server Key Exchange - 12
- Server Hello Done - 14
- Client Key Exchange - 16
- Encrypted Handshake Message

3.10

Ο πελάτης έστειλε 1 μήνυμα Client Hello. Κάθε μήνυμα Client Hello αντιστοιχεί και σε μία σύνδεση TCP.

3.11

Η μέγιστη έκδοση είναι η TLS 1.2.

3.12

Το μήκος του τυχαίου αριθμού που περιέχει είναι 32 bytes. Τα πρώτα 4 είναι `2e 1a 6e 1b`. Κανονικά τα 4 πρώτα bytes αναπαριστούν την χρονική στιγμή αποστολής (GMT Unix Time).

3.13

Υπάρχουν 17 σουίτες, και οι δύο πρώτες έχουν δεκαεξαδικές τιμές: `0x1301`, `0x1303`.

3.14

Θα χρησιμοποιηθεί η έκδοση TLS 1.2. Η σουίτα κωδίκων κρυπτογράφησης που τελικά επιλέχθηκε έχει όνομα `TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256` και δεκαεξαδική τιμή `0xc02f`.

3.15

Το μήκος σε byte του τυχαίου αριθμού που περιέχει το μήνυμα Server Hello είναι 32. Τα πρώτα 4 bytes είναι `42 63 84 18`.

3.16

Τόσο στο Client Hello όσο και στο Server Hello, το πεδίο Compression Method έχει την τιμή `null`, επομένως δεν χρησιμοποιείται συμπίεση.

3.17

Είναι:

- Αλγόριθμος ανταλλαγής κλειδιών: `ECDHE`
- Αλγόριθμος πιστοποίησης ταυτότητας: `RSA`
- Αλγόριθμος κρυπτογράφησης: `AES_128_GCM`
- Συνάρτηση κατακερματισμού: `SHA256`

3.18

Με βάση το πεδίο length της επικεφαλίδας, η εγγραφή TLS που μεταφέρει το πιστοποιητικό του εξυπηρετητή έχει μήκος 4278 bytes.

3.19

Μεταφέρονται 3 πιστοποιητικά. Τα ονόματά τους είναι:

- Let's Encrypt, R3
- Internet Security Research Group, ISRG Root X1
- Digital Signature Trust Co., DST Root CA X3

3.20

Χρειάστηκαν 4 πλαίσια Ethernet.

3.21

Το μήκος και των δύο δημοσίων κλειδιών που αποστέλλουν ο πελάτης και ο εξυπηρετητής αντίστοιχα είναι 32 bytes. Τα πρώτα 5 γράμματα του κλειδιού του πελάτη είναι `dda63`, ενώ του εξυπηρετητή είναι `fa0f9`.

3.22

Το μήκος της εγγραφής είναι 6 bytes συνολικά.

3.23

Το μήκος της εγγραφής είναι 45 bytes συνολικά.

3.24

Ναι, παρατηρήσαμε.

3.25

Ναι, παρατηρήσαμε από την πλευρά του υπολογιστή μας.

3.26

Το Encrypted Alert είναι υπεύθυνο για τον τερματισμό της σύνδεσης.

3.27

Παρατηρούμε ότι η αναζήτηση επιστρέφει αποτέλεσμα μόνο στην περίπτωση του HTTP, και όχι στην περίπτωση του HTTPS.

3.28

- Πιστοποίηση αυθεντικότητας: Επιτυγχάνεται με την χρήση των certificates
- Εμπιστευτικότητα: Επιτυγχάνεται με την κρυπτογράφηση των μηνυμάτων
- Ακεραιότητα: Επιτυγχάνεται με την χρήση των hash functions
  Αυτό έρχεται σε αντίθεση με το HTTP, όπου δεν συμβαίνει τίποτα από τα παραπάνω.