



## ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

### ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΟΝΟΜΑΤΕΠΩΝΥΜΟ: ΓΚΟΥΜΕ ΛΑΟΥΡΕΝΤΙΑΝ

ΑΜ: 031 18 014

ΕΞΑΜΗΝΟ: 7<sup>ο</sup>

ΟΜΑΔΑ: 4

MAC ADDRESS: B4-69-21-1B-6C-FF

IPv4: Άσκ1, Άσκ2, Άσκ3: 10.3.20.34

ΌΝΟΜΑ ΥΠΟΛΟΓΙΣΤΗ: LAPTOP-B2DVAJKK

ΛΕΙΤΟΥΡΓΙΚΟ ΣΥΣΤΗΜΑ: WINDOWS 10

### ΔΙΚΤΥΑ ΥΠΟΛΟΓΙΣΤΩΝ



## ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ 12: Ασφάλεια

### Άσκηση 1: Πιστοποίηση αυθεντικότητας στο πρωτόκολλο HTTP

1.1) Η απόκριση του εξυπηρετητή στο αρχικό μήνυμα HTTP τύπου GET, έχει **status code 401** και φράση **Authorization Required**.

No.	Time	Source	Destination	Protocol	Length	Info
6	0.005524	10.3.20.34	147.102.40.15	HTTP	60	GET /auth/ HTTP/1.1
10	0.014408	147.102.40.15	10.3.20.34	HTTP	240	HTTP/1.1 401 Authorization Required (text/html)
21	20.490402	10.3.20.34	147.102.40.15	HTTP	129	GET /auth/ HTTP/1.1
24	20.497426	147.102.40.15	10.3.20.34	HTTP	92	HTTP/1.1 200 OK (text/html)

1.2) Βλέπουμε τα πεδία των 2 GET αιτημάτων κατά σειρά:

6	0.005524	10.3.20.34	147.102.40.15	HTTP	60	GET /auth/ HTTP/1.1
10	0.014408	147.102.40.15	10.3.20.34	HTTP	240	HTTP/1.1 401 Authorization Required (text/html)
21	20.490402	10.3.20.34	147.102.40.15	HTTP	129	GET /auth/ HTTP/1.1
24	20.497426	147.102.40.15	10.3.20.34	HTTP	92	HTTP/1.1 200 OK (text/html)

▼ Hypertext Transfer Protocol

> GET /auth/ HTTP/1.1\r\n

Host: edu-dy.cn.ntua.gr\r\n

Connection: keep-alive\r\n

Upgrade-Insecure-Requests: 1\r\n

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/97.0.46

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,\*/\*;q=0.8,application/s

Accept-Encoding: gzip, deflate\r\n

Accept-Language: el,en;q=0.9,en-GB;q=0.8,en-US;q=0.7\r\n

> Cookie: \_ga=GA1.2.2012811597.1627986400; \_gid=GA1.2.1468818265.1641637483\r\n

\r\n

[Full request URI: http://edu-dy.cn.ntua.gr/auth/]

No.	Time	Source	Destination	Protocol	Length	Info
6	0.005524	10.3.20.34	147.102.40.15	HTTP	60	GET /auth/ HTTP/1.1
10	0.014408	147.102.40.15	10.3.20.34	HTTP	240	HTTP/1.1 401 Authorization Required (text/html)
21	20.490402	10.3.20.34	147.102.40.15	HTTP	129	GET /auth/ HTTP/1.1
24	20.497426	147.102.40.15	10.3.20.34	HTTP	92	HTTP/1.1 200 OK (text/html)

▼ Hypertext Transfer Protocol

> GET /auth/ HTTP/1.1\r\n

Host: edu-dy.cn.ntua.gr\r\n

Connection: keep-alive\r\n

Cache-Control: max-age=0\r\n

> Authorization: Basic ZWR1LWR5OnBhc3N3b3Jk\r\n

Upgrade-Insecure-Requests: 1\r\n

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/97.0.469

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,\*/\*;q=0.8,application/si

Accept-Encoding: gzip, deflate\r\n

Accept-Language: el,en;q=0.9,en-GB;q=0.8,en-US;q=0.7\r\n

> Cookie: \_ga=GA1.2.2012811597.1627986400; \_gid=GA1.2.1468818265.1641637483\r\n

\r\n

[Full request URI: http://edu-dy.cn.ntua.gr/auth/]

Παρατηρούμε πως υπάρχουν 2 επιπλέον πεδία, τα **Cache-Control** και **Authorization**.

1.3) Θα ασχοληθούμε με το πεδίο **Authorization**. Το περιεχόμενο του πεδίου αυτού σε μορφή ASCII φαίνεται παρακάτω: (**Authorization: Basic ZWR1LWR5OnBhc3N3b3Jk**)

```
Authoriz ation: B
asic ZWR 1LWR5OnB
hc3N3b3J k-.Upgra
```

Οι 2 κουκίδες στο τέλος αντιστοιχούν στο Carriage Return (/r) και Newline (/n).

**1.4)** Όπως αναμέναμε, το αποτέλεσμα της αποκωδικοποίησης είναι **edy:password**.

**1.5)** Διαπιστώνουμε πως ο μηχανισμός πιστοποίησης αυθεντικότητας που παρέχει το HTTP και βασίζεται στην κωδικοποίηση Base64 **δεν είναι καθόλου ασφαλής** και αυτό διότι αρκεί κάποιος να καταφέρει να υποκλέπτει τα πακέτα της σύνδεσης μεταξύ των 2 άκρων, καθώς μετά μπορεί εύκολα να αποκωδικοποιήσει οποιαδήποτε ευαίσθητα δεδομένα εστάλησαν κατά την επικοινωνία αυτή. Συνεπώς, εφόσον ο αποστολέας και ο “κανονικός” παραλήπτης δεν είναι οι μόνοι που μπορούν να κατανοούν το περιεχόμενο της σύνδεσης **δεν υπάρχει εμπιστευτικότητα**.

## **Άσκηση 2: Υπηρεσία SSH-Secure Shell**

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

**2.2)** Χρησιμοποιούνται οι **θύρες 22 (ssh)** από τον εξυπηρετητή και **57.576** από εμάς.

**2.3)** Η θύρα 22.

**2.4)** Φίλτρο απεικόνισης: **ssh**.

**2.5)** Με το φίλτρο **ssh.protocol** εντοπίζουμε τα μηνύματα SSH τύπου Protocol, όπως φαίνονται παρακάτω:

ssh.protocol					
No.	Time	Source	Destination	Protocol	Length Info
4	0.029904	147.102.40.15	10.3.20.34	SSHv2	103 Server: Protocol (SSH-2.0-OpenSSH_6.6.1_hpn13v11_FreeBSD-20140420)
5	0.031140	10.3.20.34	147.102.40.15	SSHv2	82 Client: Protocol (SSH-2.0-PuTTY_Release_0.60)
Frame 4: 103 bytes on wire (824 bits), 103 bytes captured (824 bits) on interface \Device\NPF_{94774A54-2827-44A9-8928-281187AC}					
Ethernet II, Src: Fortinet_da:67:b0 (04:d5:90:da:67:b0), Dst: IntelCor_1b:6c:ff (b4:69:21:1b:6c:ff)					
Internet Protocol Version 4, Src: 147.102.40.15, Dst: 10.3.20.34					
Transmission Control Protocol, Src Port: ssh (22), Dst Port: 57567 (57567), Seq: 1, Ack: 1, Len: 49					
SSH Protocol					
Protocol: SSH-2.0-OpenSSH_6.6.1_hpn13v11_FreeBSD-20140420					
[Direction: server-to-client]					

Συμπεραίνουμε πως από τον εξυπηρετητή χρησιμοποιείται η **έκδοση SSH-2.0**, η **έκδοση λογισμικού OpenSSH\_6.6.1\_hpn13v11** και στα **σχόλια εντοπίζουμε το FreeBSD-20140420**.

2.6) Με τον ίδιο τρόπο, για τον πελάτη βρίσκουμε:

- Έκδοση: **SSH-2.0**
- Λογισμικό: **PuTTY\_Release\_0.60**

2.7) Εντοπίζουμε τη λίστα με τους αλγορίθμους ανταλλαγής κλειδιών:

```
kex_algorithms string: diffie-hellman-group-exchange-sha256,diffie-hellman-group-exchange-sha1,diffie-hellman-group14-sha1,diffie-hellman-group1-sha1
```

Όπως βλέπουμε, εμφανίζονται **4 kex αλγόριθμοι**. Οι 2 πρώτοι εξ αυτών είναι οι **diffie-hellman-group-exchange-sha256** και **diffie-hellman-group-exchange-sha1**.

2.8) Στο ίδιο πακέτο με πριν, εντοπίζουμε τους **2 αλγορίθμους ssh-rsa και ssh-dss**.

2.9) Οι 2 πρώτοι αλγόριθμοι κρυπτογράφησης που υποστηρίζει ο πελάτης με κατεύθυνση client to server είναι **aes256-ctr, aes256-cbc**.

2.10) Αντίστοιχα, για τους αλγορίθμους πιστοποίησης αυθεντικότητας (mac), έχουμε τους **hmac-sha1 και hmac-sha1-96**.

2.11) Αντίστοιχα για τους αλγορίθμους συμπίεσης (compression), έχουμε τους **none και zlib**.

2.12) Βλέποντας τη λίστα αλγορίθμων ανταλλαγής κλειδιών του εξυπηρετητή και συγκρίνοντας με αυτή του πελάτη, αναμένουμε να χρησιμοποιηθεί ο αλγόριθμος **diffie-hellman-group-exchange-sha256**. Πράγματι, το επαληθεύουμε από το πεδίο **Key Exchange** όπως βλέπουμε παρακάτω.

```
Key Exchange (method:diffie-hellman-group-exchange-sha256)
```

2.13) Στο πεδίο **SSH Version 2** βλέπουμε το **encryption:aes256-ctr**. Παρατηρούμε πως είναι ο πρώτος από τη λίστα του εξυπηρετητή που υπάρχει και στη λίστα του πελάτη.

2.14) Χρησιμοποιείται ο αλγόριθμος πιστοποίησης αυθεντικότητας μηνυμάτων **hmac-sha1** κατά την κατεύθυνση πελάτης → εξυπηρετητής. Ισχύει η ίδια παρατήρηση με το 2.13.

2.15) Δε χρησιμοποιείται κανένας αλγόριθμος συμπίεσης (**none**).

2.16) Επιλέγοντας ένα εκ των δύο πακέτων **Key Exchange Init**, βλέπουμε στο πεδίο **SSH Version 2** της επικεφαλίδας **SSH Protocol** τις ζητούμενες πληροφορίες.

```
SSH Version 2 (encryption:aes256-ctr mac:hmac-sha1 compression:none)
```

2.17) Καταγράφηκαν επιπλέον οι εξής τύποι μηνυμάτων SSH:

- **Diffie-Hellman Group Exchange Request (Old)**
- **Diffie-Hellman Group Exchange Group**
- **Diffie-Hellman Group Exchange Init**
- **Server: Diffie-Hellman Group Exchange Reply, New Keys**
- **New Keys**
- **Encrypted Packet**

2.18) Παρατηρούμε πως **δε γίνεται αντιληπτό ποια πακέτα αφορούν το login και το password** στην περίπτωση του SSH και ο λόγος είναι πως τα πακέτα αυτά είναι κρυπτογραφημένα.

2.19) Αναφορικά με την ασφάλεια του SSH:

- **Πιστοποίηση αυθεντικότητας:** Έχουμε authentication μέσω public-private keys, από τις ασφαλέστερες δηλαδή μεθόδους.
- **Εμπιστευτικότητα:** Λόγω της κρυπτογράφησης, το περιεχόμενο γίνεται κατανοητό μόνο από τον εξυπηρετητή και τον πελάτη.
- **Ακεραιότητα δεδομένων:** Παρέχονται hashing αλγόριθμοι για data-integrity (MAC).

Κρίνεται, επομένως, ως μια ασφαλής επιλογή.

### **Άσκηση 3: Υπηρεσία HTTPS**

3.1) Φίλτρο σύλληψης: **host bbb2.cn.ntua.gr**.

3.2) Με το φίλτρο απεικόνισης **tcp.len==0 and ((tcp.seq==0 and tcp.ack==0) or (tcp.seq==0 and tcp.ack==1) or (tcp.seq==1 and tcp.ack==1))** εμφανίζονται όλες οι τριπλές χειραψίες που έγιναν. Εμφανίζονται 24 πακέτα, επομένως συμπεραίνουμε πως έγιναν 8 TCP συνδέσεις (φαίνονται στην παρακάτω εικόνα).

3.3) Οι συνδέσεις έγιναν στις θύρες **80 (HTTP)** και **443 (HTTPS)** του εξυπηρετητή.

3.4) **80-HTTP, 443-HTTPS**

3.5) Ανοίχτηκαν **6 συνδέσεις HTTP** και **2 συνδέσεις HTTPS**.

3.6) Χρησιμοποιούνται οι θύρες **55.498** και **55.499**.

3.7) Παρατηρούμε τα εξής πεδία:

- **Content Type (1 Byte)**
- **Version (2 Bytes)**
- **Length (2 Bytes)**

3.8) Καταγράφουμε τις παρακάτω τιμές:

- **Handshake (22)**
- **Change Cipher Spec (20)**
- **Application Data (23)**

3.9) Καταγράφουμε τους εξής τύπους μηνυμάτων χειραψίας:

- **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) Ο πελάτης έστειλε **2 Client Hello**, όσες και οι **HTTPS** συνδέσεις.

3.11) Η μέγιστη υποστηριζόμενη έκδοση είναι η **TLS 1.0 (0x0301)**.

3.12) Το μήκος του τυχαίου αριθμού είναι **32 bytes**, με τα πρώτα 4 εξ αυτών να είναι τα **44 66 b8 a4**. Τα bytes αυτά δηλώνουν το **GMT Unix Time**.

▼ Random: 4466b8a441a6c1359773bbe6433667ce1457bb106e001d0c0068ed0522e9962d

GMT Unix Time: May 14, 2006 07:57:08.000000000 Θερινή ώρα GTB

3.13) Καταγράφονται **16 Cipher Suites**, ενώ οι 2 πρώτες εξ αυτών είναι οι **0x1a1a (Reserved (GREASE))** και **0x1301 (TLS\_AES\_128\_GCM\_SHA256)**. Επειδή το Reserved (GREASE) δεν είναι στην πραγματικότητα cipher, παραθέτουμε επιπλέον το **0x1302 (TLS\_AES\_256\_GCM\_SHA384)**.

3.14) Από τον εξυπηρετητή θα χρησιμοποιηθεί η έκδοση **TLS 1.2 (0x0303)**, ενώ επιλέχθηκε η σουίτα **TLS\_ECDHE\_RSA\_WITH\_AES\_128\_GCM\_SHA256 (0xc02f)**.



**3.15)** Και εδώ περιέχονται **32 bytes** στον τυχαίο αριθμό. Τα πρώτα 4 bytes του Random είναι τα **f0 c1 0b 80**.

**3.16)** Το πεδίο Compression Method έχει τιμή **null (0)**, επομένως δε χρησιμοποιείται κάποια μέθοδος συμπίεσης.

**3.17)** Τα ζητούμενα βρίσκονται στο πεδίο Cipher Suite, το οποίο στην περίπτωσή μας έχει τιμή **TLS\_ECDHE\_RSA\_WITH\_AES\_128\_GCM\_SHA256**. Ειδικότερα, από το όνομα αυτό εξάγουμε τα εξής:

- Αλγόριθμος ανταλλαγής κλειδιών: **ECDHE**
- Αλγόριθμος πιστοποίησης ταυτότητας: **RSA**
- Αλγόριθμος κρυπτογράφησης: **AES(128bits)**
- Αλγόριθμος συνάρτησης κατακερματισμού: **SHA(256bits)**

**3.18)** Είναι **4278 bytes**, όπως φαίνεται παρακάτω.

```
▼ Transport Layer Security
  ▼ TLSv1.2 Record Layer: Handshake Protocol: Certificate
    Content Type: Handshake (22)
    Version: TLS 1.2 (0x0303)
    Length: 4278
```

**3.19)** Μεταφέρονται **3 πιστοποιητικά**. Τα ονόματά τους είναι τα εξής:

- 1) id-at-commonName = **bbb2.cn.ntua.gr**
- 2) id-at-commonName = **R3**
- 3) id-at-commonName = **ISRG**

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

**3.21)** Ο πελάτης αποστέλλει δημόσιο κλειδί μήκους **32 bytes (5 αρχικά γράμματα: 2b81a)**, **όσα bytes αποστέλλει δηλαδή και ο εξυπηρετητής (5 πρότερα γράμματα: cc76b)**.

**3.22)** Είναι **6 bytes**, όπως φαίνεται παρακάτω.

No.	Time	Source	Destination	Protocol	Length	Info
322	19.740589	10.3.20.34	147.102.40.19	TLSv...	571	Client Hello
325	19.741400	10.3.20.34	147.102.40.19	TLSv...	571	Client Hello
327	19.747843	147.102.40.19	10.3.20.34	TLSv...	1514	Server Hello
332	19.750309	147.102.40.19	10.3.20.34	TLSv...	1514	Server Hello
336	19.763150	147.102.40.19	10.3.20.34	TLSv...	890	Certificate, Server Key Exchange, Server
338	19.763760	147.102.40.19	10.3.20.34	TLSv...	890	Certificate, Server Key Exchange, Server
340	19.769594	10.3.20.34	147.102.40.19	TLSv...	147	Client Key Exchange, Change Cipher Spec,
341	19.770222	10.3.20.34	147.102.40.19	TLSv...	147	Client Key Exchange, Change Cipher Spec,

  

> Frame 340: 147 bytes on wire (1176 bits), 147 bytes captured (1176 bits) on interface \Device\NPF\_...  
 > Ethernet II, Src: IntelCor\_1b:6c:ff (b4:69:21:1b:6c:ff), Dst: Fortinet\_da:67:b0 (04:d5:90:da:67:b0)  
 > Internet Protocol Version 4, Src: 10.3.20.34, Dst: 147.102.40.19  
 > Transmission Control Protocol, Src Port: 55499 (55499), Dst Port: https (443), Seq: 518, Ack: 4933  
 > Transport Layer Security
 

- > TLSv1.2 Record Layer: Handshake Protocol: Client Key Exchange
- > **TLSv1.2 Record Layer: Change Cipher Spec Protocol: Change Cipher Spec**
- > TLSv1.2 Record Layer: Handshake Protocol: Encrypted Handshake Message

  

0040	2b 81 a6 d9 b4 7f bf 4c	a0 8c 0f 5b df e1 7e ea	+-----L---[~
0050	92 ca 7d 5d 0b c8 3f 72	34 bb ee 5e 9e 3b 0e 0a	...}]...?r 4...^;..
0060	<b>14 03 03 00 01 01</b> 16 03	03 00 28 00 00 00 00 00	<b>.....</b> ..(.....
0070	00 00 00 ad 63 7f 5e 0e	4e 35 c8 fd 7d ca 5d 2d	.....^..NS...1&

3.23) Είναι 45 bytes (40 εκ των οποίων αποτελούν το Encrypted Handshake Message).

3.24) Ναι, παρατηρήσαμε.

346	19.775902	147.102.40.19	10.3.20.34	TLSv...	312	New Session Ticket, Change Cipher Spec, Encrypted Handshake Message
347	19.775902	147.102.40.19	10.3.20.34	TLSv...	312	New Session Ticket, Change Cipher Spec, Encrypted Handshake Message

3.25) Όχι, δε παρατηρήσαμε.

3.26) Ο λόγος που θα υπήρχαν, θα ήταν προκειμένου ο πελάτης να ειδοποιήσει τον εξυπηρετητή για την απόλυση της TCP σύνδεσης.

3.27) Στη περίπτωση του HTTP βρίσκουμε πακέτο που έχει ως περιεχόμενο το περιεχόμενο της ιστοσελίδας που ζητήσαμε σε μορφή html, όπως βλέπουμε παρακάτω. Αντίθετα, στα πακέτα HTTPS δε μπορούμε να βρούμε κάποιο πακέτο αναζητώντας το String BigBlueButton και αυτό διότι η πληροφορία μεταφέρεται κρυπτογραφημένη στο https σε αντίθεση με το http.



Packet details						
Narrow & Wide		Case sensitive		String		
No.	Time	Source	Destination	Protocol	Length	Info
11	0.048905	147.102.40.19	10.3.20.34	HTTP	1208	HTTP/1.1 200 OK (text/html)
File Data: 12217 bytes						
Line-based text data: text/html (285 lines)						
<pre> \n &lt;!DOCTYPE html&gt;\n &lt;html&gt;\n   &lt;head&gt;\n     &lt;meta http-equiv="Content-Type" content="text/html; charset=utf-8"&gt;\n \n     &lt;title&gt;BigBlueButton - Open Source Web Conferencing&lt;/title&gt;\n     &lt;meta name="description" content="BigBlueButton enables universities and colleges to deliver a [truncated]    &lt;meta name="keywords" content="BigBlueButton, Open Source Web Conferencing, Distan     &lt;meta name="viewport" content="width=device-width, initial-scale=1.0" /&gt;\n   &lt;/head&gt;\n   &lt;body&gt;\n     &lt;link rel="icon" href="images/favicon.png"&gt;\n \n     &lt;link rel="stylesheet" href="css/bijou.min.css"&gt;\n     &lt;link rel="stylesheet" href="css/style.css"&gt;\n     &lt;link rel="stylesheet" href="css/font-awesome.min.css"&gt;\n     &lt;link rel="stylesheet" href="css/bbb-bootstrap.css"&gt; </pre>						

### 3.28) Συγκρίνοντας το HTTP με το HTTPS, μπορούμε να πούμε πως:

- **Πιστοποίηση αυθεντικότητας:** Στο HTTPS, όταν ένας client εκκινεί έναν “διάλυο” επικοινωνίας με έναν εξυπηρετητή, ο εξυπηρετητής επαληθεύει τη γνησιότητα του αντιστοιχίζοντας το private key του με το public key στο SSL certificate (το οποίο είναι signed από μία έμπιστη αρχή) της σελίδας που επισκεπτόμαστε. Στο HTTP δεν υπάρχει κάποια αντίστοιχη διαδικασία που να εξασφαλίζει την πιστότητα του εξυπηρετητή.
- **Εμπιστευτικότητα:** Στο HTTP τα δεδομένα στέλνονται ως plaintext, επομένως είναι άμεσα αναγνώσιμα από κάποιον που θα καταφέρει να υποκλέψει κάποια πακέτα. Αντιθέτως, το περιεχόμενο στο HTTPS είναι κρυπτογραφημένο, με αποτέλεσμα ακόμα και αν κάποιος υποκλέψει πακέτα να διαβάσει κάτι που δε βγάζει νόημα και από το οποίο δε μπορεί να εξαγάγει κάτι χρήσιμο.
- **Ακεραιότητα:** Στο HTTPS είναι αδύνατον να μεταβληθούν τα δεδομένα χωρίς αυτό να γίνει αντιληπτό από τους συμμετέχοντες στη σύνδεση. Αντιθέτως, το HTTP είναι επιρρεπές σε επιθέσεις τύπου Man-In-The-Middle, οι οποίες θα μπορούσαν να αλλοιώσουν το περιεχόμενο των πακέτων.