



**Εργαστήριο Δικτύων**  
**Τμήμα Μηχανικών Η/Υ & Πληροφορικής Πανεπιστήμιο Πατρών**

**5η Εργασία**  
**Ονοματεπώνυμο: Αγησίλαος Κολλιόπουλος**  
**Αριθμός Μητρώου: 1072803**  
**Τμήμα: HMTY**

**Ερωτήσεις κατανόησης και Εργασία για το μάθημα:**  
**Σύγχρονες Εφαρμογές Ασφάλειας Δικτύων**

5η Εργασία - Κρυπτογράφηση αρχείου.

**Μέρος Α**

- 1) Εγκαταστήστε το πακέτο openssl.
  - Ποιες εντολές χρησιμοποιήσατε?  
**sudo apt install openssl**
  - Με ποια εντολή ελέγχετε η έκδοση που έχει εγκατασταθεί?  
**openssl version**
- 2) Με ποια εντολή βρίσκω όλους τους κώδικες κρυπτογράφησης που υποστηρίζονται?  
**openssl ciphers -stdname**
- 3) Με ποια εντολή βρίσκω μόνο τους κώδικες κρυπτογράφησης που υποστηρίζουν TLSv1-3?

**openssl ciphers -s -v -tls1\_3**

```
agiskallas@debian-agis:~$ openssl ciphers -s -tls1_3 -stdname
TLS_AES_256_GCM_SHA384 - TLS_AES_256_GCM_SHA384 TLSv1.3 Kx=any A
u=any Enc=AEAD Mac=AEAD
TLS_CHACHA20_POLY1305_SHA256 - TLS_CHACHA20_POLY1305_SHA256 TLSv1.3 Kx=any A
u=any Enc=CHACHA20/POLY1305(256) Mac=AEAD
TLS_AES_128_GCM_SHA256 - TLS_AES_128_GCM_SHA256 TLSv1.3 Kx=any A
u=any Enc=AEAD Mac=AEAD
```

Figure 1: TLS1\_3 Ciphers

Cipher	TLS_AES_256_GCM_SHA384	
Protocol	TLSv1.3	
Key Exchange	any	
Authentication	any	
Encryption	AESGCM(256)	Advanced Encryption Standard in Galois/Counter Mode (AES- GCM) 256-bit key
Message Authentication Code	AEAD	
Hash	SHA-384	Secure Hash Algorithm Produces a 384-bit hash value



**Εργαστήριο Δικτύων**  
**Τμήμα Μηχανικών Η/Υ & Πληροφορικής Πανεπιστήμιο Πατρών**

Cipher	TLS_CHACHA20_POLY1305_SHA256	
Protocol	TLSv1.3	
Key Exchange	any	
Authentication	any	
Encryption	CHACHA20/POLY1305(256)	ChaCha20/Poly1305 256-bit key
Message Authentication Code	AEAD	
Hash	SHA-256	Secure Hash Algorithm Produces a 256-bit hash value

Cipher	TLS_AES_128_GCM_SHA256	
Protocol	TLSv1.3	
Key Exchange	any	
Authentication	any	
Encryption	AESGCM(128)	Advanced Encryption Standard in Galois/Counter Mode (AES- GCM) 128-bit key
Message Authentication Code	AEAD	
Hash	SHA-256	Secure Hash Algorithm Produces a 256-bit hash value

- 4) Ανατρέξτε στην σελίδα <https://ciphersuite.info/> και ελέγξτε ποιοι από αυτούς είναι ευπαθείς αλγόριθμοι (weak) και ποιοι όχι. Από την ίδια σελίδα βρείτε ποιοι είναι οι πιο ισχυροί αλγόριθμοι κρυπτογράφησης (recommended και strong) που υποστηρίζουν TLSv1-3 ή/και TLSv1-2.

Recommended	TLS_CHACHA20_POLY1305_SHA256
Recommended	TLS_AES_256_GCM_SHA384
Recommended	TLS_AES_128_GCM_SHA256

Figure 2: OpenSSL TLS1\_3 Ciphers Security



**Εργαστήριο Δικτύων**  
**Τμήμα Μηχανικών Η/Υ & Πληροφορικής Πανεπιστήμιο Πατρών**

Cipher TLSv1_3	Security
TLS_AES_128_GCM_SHA256	Recommended
TLS_AES_256_GCM_SHA384	Recommended
TLS_CHACHA20_POLY1305_SHA256	Recommended
TLS_AES_128_CCM_SHA256	Secure
TLS_AES_128_CCM_8_SHA256	Secure

Cipher TLSv1_2	Security
TLS_ECCPWD_WITH_AES_128_GCM_SHA256	Recommended
TLS_ECCPWD_WITH_AES_256_GCM_SHA384	Recommended
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256	Recommended
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384	Recommended
TLS_ECDHE_ECDSA_WITH_ARIA_128_GCM_SHA256	Recommended
TLS_ECDHE_ECDSA_WITH_ARIA_256_GCM_SHA384	Recommended
TLS_ECDHE_ECDSA_WITH_CAMELLIA_128_GCM_SHA256	Recommended
TLS_ECDHE_ECDSA_WITH_CAMELLIA_256_GCM_SHA384	Recommended
TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256	Recommended
TLS_ECDHE_PSK_WITH_AES_128_GCM_SHA256	Recommended
TLS_ECDHE_PSK_WITH_AES_256_GCM_SHA384	Recommended
TLS_ECDHE_PSK_WITH_CHACHA20_POLY1305_SHA256	Recommended
TLS_ECCPWD_WITH_AES_128_CCM_SHA256	Secure
TLS_ECCPWD_WITH_AES_256_CCM_SHA384	Secure
TLS_ECDHE_ECDSA_WITH_AES_128_CCM	Secure
TLS_ECDHE_ECDSA_WITH_AES_128_CCM_8	Secure
TLS_ECDHE_ECDSA_WITH_AES_256_CCM	Secure

- 5) Αναλύστε τον κώδικα κρυπτογράφησης “ECDHE-ECDSA-AES128-GCM-SHA256” και ειδικότερα τον τρόπο δημιουργίας και ανταλλαγής κλειδιών.



## Εργαστήριο Δικτύων Τμήμα Μηχανικών Η/Υ & Πληροφορικής Πανεπιστήμιο Πατρών

**IANA name:**

TLS\_ECDHE\_ECDSA\_WITH\_AES\_128\_GCM\_SHA256

**OpenSSL name:**

ECDHE-ECDSA-AES128-GCM-SHA256

**GnuTLS name:**

TLS\_ECDHE\_ECDSA\_AES\_128\_GCM\_SHA256

**Hex code:**

0xC0, 0x2B

**TLS Version(s):**

TLS1.2, TLS1.3

---

**Protocol:**

Transport Layer Security (TLS)

**Key Exchange:**

**PFS** Elliptic Curve Diffie-Hellman Ephemeral (ECDHE)

**Authentication:**

Elliptic Curve Digital Signature Algorithm (ECDSA)

**Encryption:**

**AEAD** Advanced Encryption Standard with 128bit key in Galois/Counter mode (AES 128 GCM)

**Hash:**

Secure Hash Algorithm 256 (SHA256)

Figure 3: ECDHE Cipher

Creating temporary key pairs for each session, that are encrypted with an elliptic curve cryptography. The keys, since they are ephemeral are only used for that session, sabotaging any attempts at stealing, and using them for a different session. After the public and private keys are created, the public keys are exchanged, and the message is decrypted with the private key and the received public key. The decrypted message is used to secure the future communications.

- 6) Ανατρέξτε στην ιστοσελίδα: <https://www.javainuse.com/aesgenerator> και κρυπτογραφήστε τον αριθμό μητρώου σας. Επιλέξτε CBC mode, 256 μέγεθος κλειδιού και τυχαίο κλειδί Secret και Initialization Vector. Ποια η έξοδος?

**96Kkzaw0Kp9KC+dvBnSssA==**

- Τι είναι κωδικοποίηση base64? Πως μετατρέπεται σε αναγνώσιμη μορφή?

Το base64 είναι μια μορφή κωδικοποίησης που χρησιμοποιείται για να μετατρέψει τα binary data σε αλληλουχίες των 24 bit τα οποία μπορούν να αναπαριστούν με 4 6-bit base64 ψηφία.



**Εργαστήριο Δικτύων**  
**Τμήμα Μηχανικών Η/Υ & Πληροφορικής Πανεπιστήμιο Πατρών**

Enter text to be Decrypted

96Kkzaw0Kp9KC+dvBnSssA==

Input Text Format: ☒Base64 ☐Hex

Select Cipher Mode of Decryption

CBC

Enter IV Used During Encryption(Optional)

encryptionIntVec

Key Size in Bits

256

Enter Secret Key used for Encryption

aesEncryptionKeyaesEncryptionKey

**Decrypt**

AES Decrypted Output (**Base64**):

MTA3MjgwMw==

**Decode to Plain Text**

1072803

Figure 4: Encryption/Decryption of Student No

- 7) Τι είναι οι κρυπτογραφικές λειτουργίες hash (cryptographic hash functions) και που/πως χρησιμοποιείτε?

Τα hashing functions είναι αλγόριθμοι που μετατρέπουν μια μεταβλητή είσοδο σε μια σταθερή έξοδο. Ακόμα για να είναι ένα hash function, cryptographic, πρέπει να πληροί κάποιες προϋποθέσεις όπως:

- Να είναι πρακτικά αδύνατο να βρεθεί ένα ζεύγος εισόδων που να έχουν την ίδια έξοδο.
- Να είναι γρήγορο και να υπολογίζεται για οποιοδήποτε μέγεθος εισόδου.
- Να είναι πρακτικά αδύνατο να βρεθεί μία είσοδος που να ταιριάζει με την έξοδο εκτός και αν η έξοδος είναι γνωστή από ένα pre-calculated dictionary (rainbow table).



## Εργαστήριο Δικτύων Τμήμα Μηχανικών Η/Υ & Πληροφορικής Πανεπιστήμιο Πατρών

- Με ποια εντολή βρίσκω ποιοι hash αλγόριθμοι υποστηρίζονται από το openssl?  
**openssl dgst -list**
- Δημιουργείτε το SHA512 hash του αριθμού μητρώου σας χρησιμοποιώντας την ιστοσελίδα: <https://emn178.github.io/online-tools/sha512.html>

**95379b10b05623326120da217673c5efe5743f0f62cf30a90b2aca0cf692ad3d8365bc0ef7c4268e6b79efde34d7ff9cb5b30f4ee2269068a34b02acad371d4**

### Μέρος Β

- 1) Ακολουθήστε **όλες** τις οδηγίες από το παρακάτω σύνδεσμο:  
<https://gist.github.com/kebman/f02fe0b1dbebzipscpc9ee1a56a7885c30f014> και κρυπτογραφήστε ένα απλό αρχείο, χρησιμοποιώντας τον αριθμό μητρώου σε κάθε αρχείο και πιο συγκεκριμένα:
  - Δημιουργία private/public κλειδιών μεγέθους 4096:  
1044545\_private.pem 1044545\_public.pem
  - Δημιουργία μηνύματος προς κρυπτογράφηση:  
1044545\_msg.txt με περιεχόμενο: Hello world: 1044545
  - Δημιουργία Hash Digest (1044545\_msg.digest.txt)
  - Κρυπτογραφημένη υπογραφή (1044545\_msg.signature.bin)
- Χρησιμοποιείτε ως έτερο public κλειδί το *kvlachos\_public.pem* που είναι στο φάκελο των εγγράφων στο eclass, στον φάκελο **“7. Advanced Encryption Standard”** (σε zip αρχείο).
- Υποβάλλετε σε zip ή tar αρχείο τα ακόλουθα στο eclass:
  - 1044545\_msg.b64
  - 1044545\_msg.digest.b64
  - 1044545\_msg.signature.b64
  - 1044545\_randomkey.enc.b64
  - 1044545\_public.pem (το δικό σας δημόσιο κλειδί)
- Κάνετε upload τα ίδια αρχεία (όχι zip!! όπως είναι) εδώ:  
[https://upatrasgr-my.sharepoint.com/:f/g/personal/kvlachos\\_upatras\\_gr/Eg8eC1LOpJlIt3VSAVqr9JsB1QO6IZdLCLm\\_hahnIZ6UoQ](https://upatrasgr-my.sharepoint.com/:f/g/personal/kvlachos_upatras_gr/Eg8eC1LOpJlIt3VSAVqr9JsB1QO6IZdLCLm_hahnIZ6UoQ)

Μετά το πέρας της καταληκτικής ημερομηνίας θα γίνει αυτόματη διόρθωση των αρχείων που υποβάλλατε και θα λάβετε αυτόματο email που θα αναφέρει είτε τα αρχεία που λείπουν είτε εάν τα αρχεία αποκρυπτογραφήθηκαν επιτυχώς.

Για όσους φοιτητές τα αρχεία αποκρυπτογραφήθηκαν σωστά, θα λάβουν 2ο email με οδηγίες για τη 2η φάση της εργασίας (αποστολή μηνύματος και κλειδιού για να αποκωδικοποιηθούν). Οδηγίες θα υπάρχουν στο email που θα σταλεί.

- Μην χάνετε το private κλειδί σας.



## Εργαστήριο Δικτύων Τμήμα Μηχανικών Η/Υ & Πληροφορικής Πανεπιστήμιο Πατρών

### Commands

Generating Private key :

```
openssl genrsa -aes256 -out 1072803_private.pem 4096
```

Generating public Key :

```
openssl rsa -in 1072803_private.pem -outform PEM -pubout -out 1072803_public.pem
```

Creating the file :

```
echo "Hello world: 1072803" > 1072803_msg.txt
```

Random Key :

```
openssl rand 64 > 1072803_randomkey.bin
```

Encrypt Message :

```
openssl enc -aes-256-cbc -salt -in 1072803_msg.txt -out 1072803_msg.bin -pass file:./1072803_randomkey.bin -pbkdf2
```

Prepare the encrypted msg.bin :

```
openssl base64 -in 1072803_msg.bin -out 1072803_msg.b64
```

Using kvlachos\_public.pem :

```
openssl pkeyutl -encrypt -inkey kvlachos_public.pem -pubin -in 1072803_randomkey.bin -out  
1072803_randomkey.enc.bin
```

```
openssl base64 -in 1072803_randomkey.enc.bin -out 1072803_randomkey.enc.b64
```

Hash digest :

```
cat 1072803_msg.txt | openssl dgst -sha256 -binary | xxd -p > 1072803_msg.digest.txt
```

```
openssl base64 -in 1072803_msg.digest.txt -out 1072803_msg.digest.b64
```

Cryptographic Signature :

```
openssl dgst -sha256 -sign 1072803_private.pem -out 1072803_msg.signature.bin 1072803_msg.digest.txt
```

```
openssl base64 -in 1072803_msg.signature.bin -out 1072803_msg.signature.b64
```

Ziping:

```
zip 1072803.zip 1072803_msg.b64 1072803_msg.digest.b64 1072803_msg.signature.b64  
1072803_randomkey.enc.b64 1072803_public.pem
```