

2.1 Pod adresem <http://127.0.0.1:2001> działa prosty serwer HTTP udostępniający 2 endpointy: <http://127.0.0.1:2001/encrypt> oraz <http://127.0.0.1:2001/submit>.

(a) Uruchom serwer za pomocą poniższego polecenia:

```
docker run -p 2001:2001 --name ex1 docker.io/mazurkatarzyna/symmetric-enc-ex1:latest
podman run -p 2001:2001 --name ex1 docker.io/mazurkatarzyna/symmetric-enc-ex1:latest
```

- (b) Wyślij request do endpointa <http://127.0.0.1:2001/encrypt> używając metody HTTP GET. Otrzymasz w odpowiedzi unikalny identyfikator sesji (`session_id`), losowe słowo do zaszyfrowania (`word`) oraz klucz szyfrujący podany jako 32-bajtowy ciąg szesnastkowy (`key_hex`).
- (c) Zaszyfruj słowo odebrane od serwera użyciu algorytmu AES-256 w trybie ECB wykorzystując odebrany od serwera klucz szyfrujący.
- (d) Po wykonaniu szyfrowania, zaszyfrowany ciąg znaków zakoduj w formacie `base64`.
- (e) Po uzyskaniu zaszyfrowanego i zakodowanego wyniku wyślij go do serwera poprzez endpoint <http://127.0.0.1:2001/submit> używając metody HTTP POST, w którym przekażesz zarówno identyfikator sesji (jako `session_id`), jak i zaszyfrowany ciąg w formacie `base64` (jako `encrypted_b64`). Serwer w odpowiedzi zwróci odpowiednią informację o sukcesie lub błędzie.

UWAGI:

- Aby zaszyfrować wylosowane przez serwer słowo, wykorzystaj narzędzie `OpenSSL`.
- Aby wysłać odpowiedź do serwera, użyj narzędzia `cURL`. Pamiętaj o dodaniu nagłówka protokołu HTTP `Content-Type: application/json`.

2.2 Pod adresem <http://127.0.0.1:2002> działa prosty serwer HTTP udostępniający 2 endpointy: <http://127.0.0.1:2002/decrypt> oraz <http://127.0.0.1:2002/submit>.

(a) Uruchom serwer za pomocą poniższego polecenia:

```
docker run -p 2002:2002 --name ex2 docker.io/mazurkatarzyna/symmetric-enc-ex2:latest
podman run -p 2002:2002 --name ex2 docker.io/mazurkatarzyna/symmetric-enc-ex2:latest
```

- (b) Wyślij request do endpointa <http://127.0.0.1:2002/decrypt> używając metody HTTP GET. Otrzymasz w odpowiedzi unikalny identyfikator sesji (`session_id`), zakodowane i zaszyfrowane słowo (`encrypted_b64`) oraz klucz deszyfrujący podany jako 32-bajtowy ciąg szesnastkowy (`key_hex`).
- (c) Odkoduj otrzymane słowo, stosując kodowanie `base64`. Następnie odszyfruj odkodowane słowo algorytmem AES-256 w trybie ECB z użyciem podanego klucza deszyfrującego.
- (d) Po uzyskaniu odszyfrowanego i odkodowanego wyniku wyślij go do serwera poprzez endpoint <http://127.0.0.1:2002/submit> używając metody HTTP POST, w którym przekażesz zarówno identyfikator sesji (jako `session_id`), jak i odkodowane i odszyfrowane słowo (jako `decrypted_word`). Serwer w odpowiedzi zwróci odpowiednią informację o sukcesie lub błędzie.

UWAGI:

- Aby odszyfrować zaszyfrowane przez serwer słowo, wykorzystaj narzędzie `OpenSSL`.
- Aby wysłać odpowiedź do serwera, użyj narzędzia `cURL`. Pamiętaj o dodaniu nagłówka protokołu HTTP `Content-Type: application/json`.

2.3 Pod adresem <http://127.0.0.1:2003> działa prosty serwer HTTP udostępniający 2 endpointy: <http://127.0.0.1:2003/encrypt> oraz <http://127.0.0.1:2003/submit>.

- (a) Uruchom serwer za pomocą poniższego polecenia:

```
docker run -p 2003:2003 --name ex3 docker.io/mazurkatarzyna/symmetric-enc-ex3:latest
podman run -p 2003:2003 --name ex3 docker.io/mazurkatarzyna/symmetric-enc-ex3:latest
```

- (b) Wyślij request do endpointa <http://127.0.0.1:2003/encrypt> używając metody HTTP GET. Otrzymasz w odpowiedzi unikalny identyfikator sesji (`session_id`), losowe słowo do zaszyfrowania (`word`) oraz klucz szyfrujący podany jako 32-bajtowy ciąg szesnastkowy (`key_hex`).
- (c) Zaszyfruj otrzymane od serwera słowo, stosując algorytm CAMELLIA-128 w trybie ECB wykorzystując odebrany od serwera klucz szyfrujący.
- (d) Po wykonaniu szyfrowania, zaszyfrowany ciąg znaków zakoduj w formacie `base64`.
- (e) Po uzyskaniu zaszyfrowanego i zakodowanego wyniku wyślij go do serwera poprzez endpoint <http://127.0.0.1:2003/submit> używając metody HTTP POST, w którym przekażesz zarówno identyfikator sesji (jako `session_id`), jak i zaszyfrowany ciąg w formacie `base64` (jako `encrypted_b64`). Serwer w odpowiedzi zwróci odpowiednią informację o sukcesie lub błędzie.

UWAGI:

- Aby zaszyfrować wylosowane przez serwer słowo, wykorzystaj narzędzie `OpenSSL`.
- Aby wysłać odpowiedź do serwera, użyj narzędzia `cURL`. Pamiętaj o dodaniu nagłówka protokołu HTTP `Content-Type: application/json`.

2.4 Pod adresem <http://127.0.0.1:2004> działa prosty serwer HTTP udostępniający 2 endpointy: <http://127.0.0.1:2004/decrypt> oraz <http://127.0.0.1:2004/submit>.

- (a) Uruchom serwer za pomocą poniższego polecenia:

```
docker run -p 2004:2004 --name ex4 docker.io/mazurkatarzyna/symmetric-enc-ex4:latest
podman run -p 2004:2004 --name ex4 docker.io/mazurkatarzyna/symmetric-enc-ex4:latest
```

- (b) Wyślij request do endpointa <http://127.0.0.1:2004/decrypt> używając metody HTTP GET. Otrzymasz w odpowiedzi unikalny identyfikator sesji (`session_id`), zakodowane i zaszyfrowane słowo (`encrypted_b64`) oraz klucz deszyfrujący podany jako 32-bajtowy ciąg szesnastkowy (`key_hex`).
- (c) Odkoduj otrzymane słowo, stosując kodowanie `base64`. Następnie odszyfruj odkodowane słowo algorytmem CAMELLIA-128 w trybie ECB z użyciem podanego klucza deszyfrującego.
- (d) Po uzyskaniu odszyfrowanego i odkodowanego wyniku wyślij go do serwera poprzez endpoint <http://127.0.0.1:2004/submit> używając metody HTTP POST, w którym przekażesz zarówno identyfikator sesji (jako `session_id`), jak i odkodowane i odszyfrowane słowo (jako `decrypted_word`). Serwer w odpowiedzi zwróci odpowiednią informację o sukcesie lub błędzie.

UWAGI:

- Aby odszyfrować zaszyfrowane przez serwer słowo, wykorzystaj narzędzie `OpenSSL`.
- Aby wysłać odpowiedź do serwera, użyj narzędzia `cURL`. Pamiętaj o dodaniu nagłówka protokołu HTTP `Content-Type: application/json`.

2.5 Pod adresem <http://127.0.0.1:2005> działa prosty serwer HTTP udostępniający 2 endpointy: <http://127.0.0.1:2005/encrypt> oraz <http://127.0.0.1:2005/submit>.

- (a) Uruchom serwer za pomocą poniższego polecenia:

```
docker run -p 2005:2005 --name ex5 docker.io/mazurkatarzyna/symmetric-enc-ex5:latest
podman run -p 2005:2005 --name ex5 docker.io/mazurkatarzyna/symmetric-enc-ex5:latest
```

- (b) Wyślij request do endpointa <http://127.0.0.1:2005/encrypt> używając metody HTTP GET. Otrzymasz w odpowiedzi unikalny identyfikator sesji (`session_id`) oraz losowe słowo do zaszyfrowania (`word`).
- (c) Wygeneruj klucz szyfrujący (klucz musi mieć 192 bity).
- (d) Zaszyfruj otrzymane od serwera słowo, stosując algorytm ARIA-192 w trybie ECB wykorzystując wygenerowany przez Ciebie (w punkcie 2.5c) klucz szyfrujący.
- (e) Po wykonaniu szyfrowania zaszyfrowany ciąg znaków zakoduj w formacie `base64`.
- (f) Po uzyskaniu zaszyfrowanego i zakodowanego wyniku wyślij go do serwera poprzez endpoint <http://127.0.0.1:2005/submit> używając metody HTTP POST, w którym przekażesz: identyfikator sesji (`session_id`), zaszyfrowany ciąg w formacie `base64` (`encrypted_b64`), wygenerowany przez Ciebie klucz szyfrujący w formacie szesnastkowym (`key_hex`, 192 bity). Serwer w odpowiedzi zwróci odpowiednią informację o sukcesie lub błędzie.

UWAGI:

- Aby zaszyfrować wylosowane przez serwer słowo i wygenerować klucz szyfrujący wykorzystaj narzędzie `OpenSSL`.
- Aby wysłać odpowiedź do serwera, użyj narzędzia `cURL`. Pamiętaj o dodaniu nagłówka protokołu HTTP `Content-Type: application/json`.

2.6 Pod adresem <http://127.0.0.1:2006> działa prosty serwer HTTP udostępniający 2 endpointy: <http://127.0.0.1:2006/encrypt> oraz <http://127.0.0.1:2006/submit>.

- (a) Uruchom serwer za pomocą poniższego polecenia:

```
docker run -p 2006:2006 --name ex6 docker.io/mazurkatarzyna/symmetric-enc-ex6:latest
podman run -p 2006:2006 --name ex6 docker.io/mazurkatarzyna/symmetric-enc-ex6:latest
```

- (b) Wyślij żądanie HTTP GET do endpointa <http://127.0.0.1:2006/encrypt>. Otrzymasz w odpowiedzi unikalny identyfikator sesji (`session_id`) oraz losowe słowo do zaszyfrowania (`word`).
- (c) Wygeneruj klucz szyfrujący (klucz musi mieć 128 bitów).
- (d) Wygeneruj wektor inicjalizacyjny (IV) (klucz musi mieć 128 bitów).
- (e) Zaszyfruj pobrane od serwera słowo przy użyciu algorytmu AES-128 w trybie CBC, z kluczem szyfrującym oraz wektorem inicjalizującym (IV) wygenerowanymi przez Ciebie (w punktach 2.6c i 2.6d).
- (f) Po wykonaniu szyfrowania zakoduj zaszyfrowany ciąg do formatu `base64`.
- (g) Po uzyskaniu zaszyfrowanego i zakodowanego wyniku wyślij go do serwera poprzez endpoint <http://127.0.0.1:2006/submit> używając metody HTTP POST, przekazując następujące dane: identyfikator sesji (`session_id`), zaszyfrowany ciąg w formacie `base64` (`encrypted_b64`), klucz szyfrujący w formacie szesnastkowym (`key_hex`, 128 bitów) oraz IV w formacie szesnastkowym (`iv_hex`, 128 bitów). Serwer w odpowiedzi zwróci odpowiednią informację o sukcesie lub błędzie.

UWAGI:

- Aby zaszyfrować wylosowane przez serwer słowo, wygenerować klucz szyfrujący oraz IV, wykorzystaj narzędzie `OpenSSL`.
- Aby wysłać odpowiedź do serwera, użyj narzędzia `cURL`. Pamiętaj o dodaniu nagłówka protokołu `HTTP Content-Type: application/json`.

2.7 Pod adresem <http://127.0.0.1:2007> działa prosty serwer HTTP udostępniający 2 endpointy: <http://127.0.0.1:2007/decrypt> oraz <http://127.0.0.1:2007/submit>.

(a) Uruchom serwer za pomocą poniższego polecenia:

```
docker run -p 2007:2007 --name ex7 docker.io/mazurkatarzyna/symmetric-enc-ex7:latest
podman run -p 2007:2007 --name ex7 docker.io/mazurkatarzyna/symmetric-enc-ex7:latest
```

- (b) Wyślij żądanie HTTP GET do endpointa <http://127.0.0.1:2007/decrypt>. W odpowiedzi otrzymasz identyfikator sesji (`session_id`), zakodowany w formacie `base64` ciąg zaszyfrowanego tekstu (`encrypted_b64`), użyte hasło (`password`) oraz IV w formacie szesnastkowym (`iv_hex`). Serwer zaszyfrował losowo wybrane słowo przy użyciu algorytmu AES-256-CBC, z kluczem wygenerowanym na podstawie hasła przy użyciu funkcji PBKDF2, oraz losowego wektora inicjalizującego (IV).
- (c) Odkoduj otrzymane słowo, stosując kodowanie `base64`. Następnie odszyfruj odkodowane słowo algorytmem AES-256-CBC, z użyciem kluczem wygenerowanym na podstawie hasła przy użyciu funkcji PBKDF2, oraz losowego wektora inicjalizującego (IV).
- (d) Pamiętaj, że do szyfrowania NIE użyto soli, więc w deszyfrowaniu również NIE JEST potrzebna.
- (e) Wyślij do serwera odpowiedź poprzez endpoint <http://127.0.0.1:2007/submit>, używając metody HTTP POST, przekazując dane w formacie JSON: identyfikator sesji (`session_id`) oraz odszyfrowane słowo (`decrypted_word`). Serwer w odpowiedzi zwróci odpowiednią informację o sukcesie lub błędzie.

UWAGI:

- Aby odszyfrować zaszyfrowane przez serwer słowo, wykorzystaj narzędzie `OpenSSL`.
- Aby wysłać odpowiedź do serwera, użyj narzędzia `cURL`. Pamiętaj o dodaniu nagłówka protokołu `HTTP Content-Type: application/json`.

2.8 Pod adresem <http://127.0.0.1:2008> działa prosty serwer HTTP udostępniający 2 endpointy: <http://127.0.0.1:2008/decrypt> oraz <http://127.0.0.1:2008/submit>.

(a) Uruchom serwer za pomocą poniższego polecenia:

```
docker run -p 2008:2008 --name ex8 docker.io/mazurkatarzyna/symmetric-enc-ex8:latest
podman run -p 2008:2008 --name ex8 docker.io/mazurkatarzyna/symmetric-enc-ex8:latest
```

- (b) Wyślij żądanie HTTP GET do endpointa <http://127.0.0.1:2008/decrypt>. W odpowiedzi otrzymasz identyfikator sesji (`session_id`), zakodowany w formacie `base64` ciąg zaszyfrowane, losowe słowo (`encrypted_b64`), użyte hasło (`password`) oraz IV w formacie szesnastkowym (`iv_hex`). Serwer zaszyfrował losowo wybrane słowo przy użyciu algorytmu 3DES, z kluczem wygenerowanym na podstawie hasła przy użyciu funkcji PBKDF2, oraz losowego wektora inicjalizującego (IV).
- (c) Odkoduj otrzymane słowo, stosując kodowanie `base64`. Następnie, odszyfruj otrzymany ciąg, stosując algorytm 3DES, funkcję PBKDF2 do wyprowadzenia klucza z hasła oraz przekazany IV.
- (d) Pamiętaj, że do szyfrowania NIE użyto soli, więc w deszyfrowaniu również NIE JEST potrzebna.
- (e) Wyślij do serwera odpowiedź poprzez endpoint <http://127.0.0.1:2008/submit>, używając metody HTTP POST, przekazując dane w formacie JSON: identyfikator sesji (`session_id`) oraz odszyfrowane słowo (`decrypted_word`). Serwer w odpowiedzi zwróci odpowiednią informację o sukcesie lub błędzie.

UWAGI:

- Aby odszyfrować zaszyfrowane przez serwer słowo, wykorzystaj narzędzie `OpenSSL`.
- Aby wysłać odpowiedź do serwera, użyj narzędzia `cURL`. Pamiętaj o dodaniu nagłówka protokołu `HTTP Content-Type: application/json`.

2.9 Pod adresem <http://127.0.0.1:2009> działa prosty serwer HTTP udostępniający 2 endpointy: <http://127.0.0.1:2009/decrypt> oraz <http://127.0.0.1:2009/submit>.

(a) Uruchom serwer za pomocą poniższego polecenia:

```
docker run -p 2009:2009 --name ex9 docker.io/mazurkatarzyna/symmetric-enc-ex9:latest  
podman run -p 2009:2009 --name ex9 docker.io/mazurkatarzyna/symmetric-enc-ex9:latest
```

- (b) Wządanie HTTP GET do endpointa <http://127.0.0.1:2009/decrypt>. W odpowiedzi otrzymasz identyfikator sesji (`session_id`), zakodowany w formacie `base64` ciąg zaszyfrowane, losowe słowo (`encrypted_b64`) oraz użyte hasło (`password`). Serwer zaszyfrował losowo wybrane słowo przy użyciu algorytmu AES-256-ECB, z kluczem wygenerowanym na podstawie hasła przy użyciu funkcji PBKDF2 z liczbą iteracji równą 356.
- (c) Odkoduj otrzymane słowo, stosując kodowanie `base64`. Następnie, odszyfruj otrzymany ciąg, stosując algorytm AES-256-ECB, z kluczem wygenerowanym na podstawie hasła przy użyciu funkcji PBKDF2 z liczbą iteracji równą 356.
- (d) Pamiętaj, że do szyfrowania NIE użyto soli, więc w deszyfrowaniu również NIE JEST potrzebna.
- (e) Wyślij do serwera odpowiedź poprzez endpoint <http://127.0.0.1:2009/submit>, używając metody HTTP POST, przekazując dane w formacie JSON: identyfikator sesji (`session_id`) oraz odszyfrowane słowo (`decrypted_word`).

UWAGI:

- Aby odszyfrować zaszyfrowane przez serwer słowo, wykorzystaj narzędzie `OpenSSL`.
- Aby wysłać odpowiedź do serwera, użyj narzędzia `cURL`. Pamiętaj o dodaniu nagłówka protokołu `HTTP Content-Type: application/json`.

2.10 Pod adresem <http://127.0.0.1:2010> działa prosty serwer HTTP udostępniający 2 endpointy: <http://127.0.0.1:2010/decrypt> oraz <http://127.0.0.1:2010/submit>.

(a) Uruchom serwer za pomocą poniższego polecenia:

```
docker run -p 2010:2010 --name ex10 docker.io/mazurkatarzyna/symmetric-enc-ex10:latest  
podman run -p 2010:2010 --name ex10 docker.io/mazurkatarzyna/symmetric-enc-ex10:latest
```

- (b) Wyślij żądanie HTTP GET do endpointa <http://127.0.0.1:2010/decrypt>. W odpowiedzi otrzymasz identyfikator sesji (`session_id`), zakodowany w formacie `base64` ciąg zaszyfrowanego tekstu (`encrypted_b64`), klucz użyty do szyfrowania (`key_hex`) oraz IV (jeśli wylosowany algorytm go wymaga, `iv_hex`).
- (c) Twoim zadaniem jest odszyfrowanie zaszyfrowanego słowa przesłanego przez serwer. Serwer szyfruje losowo wybrane słowo przy użyciu losowego algorytmu dostępnego w bibliotece `OpenSSL`.
- (d) Odgadnij, jaki algorytm został wykorzystany do zaszyfrowania słowa, i odszyfruj otrzymany ciąg. (Pamiętaj, że po zaszyfrowaniu ciąg był zakodowany przy użyciu kodowania `base64`, więc najpierw odkoduj, a później odszyfruj ciąg.)
- (e) Wyślij do serwera odpowiedź poprzez endpoint <http://127.0.0.1:2010/submit>, używając metody HTTP POST, przekazując dane w formacie JSON: identyfikator sesji (`session_id`), odszyfrowane słowo (`decrypted_word`) i znaleziony algorytm, którym słowo zostało zaszyfrowane (`algorithm`).

UWAGI:

- Aby odszyfrować zaszyfrowane przez serwer słowo, wykorzystaj narzędzie `OpenSSL`.
- Aby wysłać odpowiedź do serwera, użyj narzędzia `cURL`. Pamiętaj o dodaniu nagłówka protokołu `HTTP Content-Type: application/json`.

2.11 Pod adresem <http://127.0.0.1:2011> działa prosty serwer HTTP udostępniający 2 endpointy: <http://127.0.0.1:2011/encrypt> oraz <http://127.0.0.1:2011/submit>.

(a) Uruchom serwer za pomocą poniższego polecenia:

```
docker run -p 2011:2011 --name ex11 docker.io/mazurkatarzyna/symmetric-enc-ex11:latest  
podman run -p 2011:2011 --name ex11 docker.io/mazurkatarzyna/symmetric-enc-ex11:latest
```

- (b) Wyślij żądanie HTTP GET do endpointa <http://127.0.0.1:2011/encrypt>. W odpowiedzi otrzymasz identyfikator sesji (`session_id`), obrazek do zaszyfrowania (`image_data_base64`), klucz do szyfrowania (`key_hex`) oraz IV (`iv`).
- (c) Do zaszyfrowania obrazka użyj algorytmu `ARIA-128-CTR` (bez paddingu), wykorzystując odebrany od serwera klucz szyfrujący oraz IV.
- (d) Wyślij do serwera odpowiedź poprzez endpoint <http://127.0.0.1:2011/submit>, używając metody HTTP POST, przekazując dane w formacie JSON: identyfikator sesji (`session_id`) i zaszyfrowany obrazek zakodowany w `base64` (obrazek jest najpierw zaszyfrowany, a potem zakodowany) (jako `encrypted_data`). Serwer zwróci w odpowiedzi informację o tym, czy obrazek jest prawidłowo zaszyfrowany.

UWAGI:

- Aby zaszyfrować wylosowany przez serwer obrazek, wykorzystaj narzędzie `OpenSSL`.
- Aby wysłać odpowiedź do serwera, użyj narzędzia `cURL`. Pamiętaj o dodaniu nagłówka protokołu `HTTP Content-Type: application/json`.

2.12 Pod adresem <http://127.0.0.1:2012> działa prosty serwer HTTP udostępniający 2 endpointy: <http://127.0.0.1:2012/decrypt> oraz <http://127.0.0.1:2012/submit>.

(a) Uruchom serwer za pomocą poniższego polecenia:

```
docker run -p 2012:2012 --name ex12 docker.io/mazurkatarzyna/symmetric-enc-ex12:latest  
podman run -p 2012:2012 --name ex12 docker.io/mazurkatarzyna/symmetric-enc-ex12:latest
```

- (b) Wyślij żądanie HTTP GET do endpointa <http://127.0.0.1:2012/decrypt>. W odpowiedzi otrzymasz identyfikator sesji (`session_id`), obrazek do odszyfrowania (`image_data_base64`), klucz do odszyfrowania (`key_hex`) oraz IV (`iv`).
- (c) Do odszyfrowania obrazka użyj algorytmu `ARIA-256-CFB` (bez paddingu), wykorzystując odebrany od serwera klucz deszyfrujący oraz IV.
- (d) Wyślij do serwera odpowiedź poprzez endpoint <http://127.0.0.1:2012/submit>, używając metody HTTP POST, przekazując dane w formacie JSON: identyfikator sesji (`session_id`) i odszyfrowany obrazek zakodowany w `base64` (obrazek jest najpierw odszyfrowany, a potem zakodowany) (jako `decrypted_data`). Serwer zwróci w odpowiedzi informację o tym, czy obrazek jest prawidłowo odszyfrowany.
- (e) Zapisz odszyfrowany obrazek na dysku z rozszerzeniem `*.png`.

UWAGI:

- Aby odszyfrować wylosowany przez serwer obrazek, wykorzystaj narzędzie `OpenSSL`.
- Aby wysłać odpowiedź do serwera, użyj narzędzia `cURL`. Pamiętaj o dodaniu nagłówka protokołu `HTTP Content-Type: application/json`.