# Information systems security
# Cryptographic algorithms

The exercise aims to teach you to work with the OpenSSL program that allows using different cryptographic algorithms and get acquainted with the differences between individual algorithms. In the exercise, it is necessary to answer all the questions asked. We will use a Kali Linux VM for the exercise.

**Question:** Run `openssl` with the version parameter. Which version of OpenSSL is installed on your computer?
**Answer**:_____

**Question:** Review the cryptographic algorithms supported by OpenSSL with the `openssl list -cipher-commands` command. Write the main families (AES, DES, ...) of supported algorithms (name the families you recognise):
**Odgovor**:_____

## 1. File encryption and decryption

OpenSSL allows file encryption with arbitrary algorithms. To solve this exercise, first create a document in the `/tmp` directory:

1) Create a text document named `cleartext.txt` with the following contents:
   `This is <name and surname> cryptographic exercise`
   Command to use: `echo "THE ABOVE TEXT!" > /tmp/cleartext.txt`
   Make sure to replace the "THE ABOVE TEXT!" with the text above ;).

Encrypt the file with AES-192-CBC, BASE64 and DES algorithms by using the OpenSSL program (the following are the commands for each of the above algorithms):

```
openssl enc -aes-192-cbc -in /tmp/cleartext.txt -out /tmp/encrypted_aes_192_cbc.enc
openssl enc -base64 -in /tmp/cleartext.txt -out /tmp/encoded_base64.enc
openssl enc -des -in /tmp/cleartext.txt -out /tmp/encrypted_des.enc
```

Write down the secret key you used for encryption: _____
Which algorithm doesn't need a secret key? _____
Write down the above algorithms according to strength, from strongest to weakest:
_____

Look at the encrypted files (cat command) and describe them (how do they look? Which characters are within? Is there anything interesting at the beginning of the files? etc.):
_____
_____

Decrypt files using OpenSSL
`openssl enc -d -aes-192-cbc -in /tmp/encrypted_aes_192_cbc.enc`

Write down the commands for decrypting DES and BASE64 encoded files:
_____
_____

What happens when you try to decrypt with a wrong decryption key (password)?
_____

What happens if you try to decrypt by using the wrong algorithm?

_____

Create MD5 and SHA1 checksum of the file cleartext.txt:
`openssl dgst –md5 /tmp/cleartext.txt`
`openssl dgst –sha1 /tmp/cleartext.txt`
What is the difference between checksums (is there a difference in the number of characters?)?

_____

Both protocols are outdated. Find the command on the Internet to create one of the hashing protocols supported today. What is the command to create a SHA2 256 bits hash? Which command can be used to create SHA3 256 bits hash?

_____

What is the command to find out all supported digest (hashing) algorithms
HINT: check the beginning of the exercise when you learned the supported cryptographic algorithms – it's similar

_____

## 2. Password "encryption"

OpenSSL allows the encryption of passwords in the so-called crypt-style previously used by Unix and Linux servers. To encrypt passwords, the passwd parameter is used. In addition to crypt-style, OpenSSL can encrypt passwords in MD5 style and using other advanced algorithms such as SHA512, bcrypt etc., which are currently used by modern Linux OS distributions.
Encrypt password in crypt style:
`openssl passwd <<password>>`
Select a random password (instead of `<<password>>`, enter the desired password. Write down the string you received as a result: _____

Repeat the command with the same password. Did you receive the same result?

_____

You should have, but you didn't because the OpenSSL, by default, adds two random characters as salt. So what is salt, and what is it used for when hashing the password?

_____
_____
_____

Add your salt (only two characters because the OpenSSL will ignore the rest):
Choose the same password as above, and add the salt in front of it. Repeat the process twice. This time the hash should be the same both times. Write down the resulting hash. (replace `<<salt>>` with your own, and `<<passwd>>` with the password as above):
`openssl passwd –salt <<salt>> <<password>>`

_____

The first two characters are salt. Take the salt from one of the results where OpenSSL automatically added the salt. You should have the same resulting hash. TRY IT!

Repeat the exercise by creating MD5 hashes instead of crypt hashes:
`openssl passwd -1 <<password>>`

`openssl passwd -1 –salt <salt> <<password>>`
The MD5 password hash is identified with the first two characters in the result. What is it?

_____

Generate SHA256 and SHA512 hashes. Write down the command used and the hash type identifiers (first two characters of the resulting hash.

_____
_____
_____
_____

## 3. Prime numbers testing

OpenSSL allows testing the prime numbers:

`openssl prime <number>`

Test whether the number 119054759245460753 is the prime number. Is it?

_____

Test the number 101. Is it prime? _____
What kind of format is the number tested by OpenSSL (Binary? Octal? Decimal? Hexadecimal?) Please observe that the OpenSSL changes the number format before testing it!

_____

## 4. Encryption speed test

OpenSSL allows the encryption speed test with the speed parameter.
`openssl speed <alghoritm>`

Example:
`openssl speed rsa`

Check the speed of the following algorithms:
RSA (write down the result of the test for `2048 bits, sign/s`): _____
RSA (write down the result of the test for `4096 bits, sign/s`): _____

Write down the command to test the DES algorithm speed

_____

For DES (write down the result of the test for `des cbc, 64 bytes`): _____

Write down the command to test the AES algorithm speed

_____

For AES (write down the result of the test for `192-cbc, 256 bytes`):

_____

## 5. Generate asymmetric pair of keys

OpenSSL allows the generation of public and private key pairs for asymmetric encryption (more on the PKI and asymmetric encryption next time).

`openssl genrsa –out /tmp/mykey.pem 4096`

View the file `mykey.pem` (with `cat` command).
How does the file `mykey.pem` looks like? (which characters are used? What do you think which algorithms are used to create this text? To which encryption from the first exercise today it is more similar?)

_____

_____

In the file `mykey.pem`, there is the public and private key.
You can extract the public key with the following command:

```
openssl rsa –in /tmp/mykey.pem –pubout
```

What does it look like? (which characters are used? What do you think which algorithms are used to create this text? To which encryption from the first exercise today it is more similar?)

_____

_____

Write down the command to extract the public key and the exponent
HINT: add modulus switch :)

_____

Write down the command to extract the private key.

_____