Security and Privacy

Lab Worksheets

João Vilela and Manuel E. Correia

— PL 2: Symmetric Cryptography + Hash Functions & MACs

(Based on materials from Manuel Barbosa and Bernardo Portela)

- OpenSSL manpage: https://www.openssl.org/docs/man1.1.1/man1/
- Cryptography hazmat python manual: https://cryptography.io/en/latest/hazmat/primitives/
- 0 - Start by installing the cryptography library for python
  - in case you do not have it yet: `pip install cryptography --user`
- You should have the cryptography library version 3.3
  - (check in python by: `import cryptography; help(cryptography)`)
- (if you do not have python 3 installed, you can install the anaconda distribution, which includes all packages required)

1 - We can use the python cryptography library (`pyca/cryptography`) to compute one block with AES.

```python
from os import urandom
from binascii import hexlify
from cryptography.hazmat.primitives.ciphers import Cipher, algorithms, modes
key = urandom(16)
iv = urandom(16)
cipher = Cipher(algorithms.AES(key), modes.ECB())
encryptor = cipher.encryptor()
# What happens if you don't pass 16-byte input?
ct = encryptor.update(b"attack at dawn!!") + encryptor.finalize()
print(hexlify(key))
cphFile = open("ciphertext.bin", "wb")
cphFile.write(ct)
cphFile.close()
```

2 - We can use openSSL in the console to invert the block.

```
openssl enc -aes-128-ecb -nopad -d -K <key_in_hex> -in ciphertext.bin
```

(You should replace `<key_in_hex>` by the key printed by the above python program.)

Can you check if inversion was correct?

Study and understand the options used in this command (run openssl, and then `help aes-128-ecb`).

3 - Modify the python example above to encrypt a file in CBC mode.

4 - Decrypt the file with OpenSSL and check for success.

5 - Edit the file to change the value of (but not delete!) one byte and decrypt again. You can use `ghex` editor in Linux or equivalent.

- What happened?
- Could you recover a file encrypted with CBC if the IV and the first ciphertext block were corrupted or lost?
- Could you recover it if during a satellite transmission one bit of the ciphertext is not delivered?
- Could you modify a byte in the middle of a CBC encrypted file without fully re-encrypting it?

6 - Repeat the exercise with CTR mode. What are the differences?

7 - Use OpenSSL to compute the SHA-256 hash value of a file (it can be the pdf file for this Lab assignments) and then write a Python program that recomputes the same SHA-256 for the same file.

8 - Use the tool available here (or any other tool that works) to construct two PDFs with the same SHA-1 value. One of the PDFs should explain how a single SHA-1 collision allows finding infinite pairs of colliding PDFs.

9 - Use openSSL to authenticate a file using HMAC

- HINT: The openSSL command is `dgst`
- Confirm that the MAC changes if you change the file and recompute it

10 - Write a Python program that recomputes the same MAC and check compatibility with openSSL