**Key extraction from Images**

Implementation:

Extracting String from image

Input: Image of any size.

Procedure: convert the rgb image to gray image and set the threshold. Resize the image to required dimension and store the binary values of the image based on the threshold.

im\_rgb = cv2.imread('download.png')  
im\_gray = cv2.cvtColor(im\_rgb, cv2.COLOR\_BGR2GRAY)  
thresh = 127  
im\_bw = cv2.threshold(im\_gray, thresh, 255, cv2.THRESH\_BINARY)[1]  
dim = (10, 10)  
# resize image  
im\_bw = cv2.resize(im\_bw, dim, interpolation=cv2.INTER\_AREA)  
img\_str = ""  
cv2.imwrite("resize.png", im\_bw)  
for i in range(0, 10):  
 for j in range(0, 10):  
 if im\_bw[i][j] <=127:  
 img\_str += "0"  
 elif im\_bw[i][j] >127:  
 img\_str += "1"

Extract the key from HMAC sha256 generator and store in a file.

def secretkey\_generate(key, message):  
 byte\_key = binascii.unhexlify(key)  
 print("length of key "+str(len(byte\_key)))  
 message = message.encode()  
 print("length of message "+str(len(message)))  
 return hmac.new(byte\_key, message, hashlib.sha256).hexdigest()

file1 = open("RandomKey.key", "w")  
file1.writelines(secret\_key)  
file1.close()

Extract the key from string and HMAC key.

def HMAC\_SHA256key(key, message):  
 final\_key = key.encode('utf-8')  
 message = message.encode()  
 return hmac.new(final\_key, message, hashlib.sha256).hexdigest()

Testing HMAC sha256 python wi:

#test case 1

secret\_key = secretkey\_generate(“0b0b0b0b0b0b0b0b0b0b0b0b0b0b0b0b0b0b0b0b","Hi There")

**length of key 20**

**length of message 8**

**secret key generated : b0344c61d8db38535ca8afceaf0bf12b881dc200c9833da726e9376c2e32cff7**

#test case 2

secret\_key = secretkey\_generate("4a656665","what do you want for nothing?")

**length of key 4**

**length of message 28**

**secret key generated : 5bdcc146bf60754e6a042426089575c75a003f089d2739839dec58b964ec3843**

#test case 3

secret\_key = secretkey\_generate("aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa","dddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddd")

**length of key 20**

**length of message 100**

**secret key generated : d50104c51b8da105ac76011e4d91fc227c4c134c0234e571bc0712da9158da9e**

#test case 4

secret\_key = secretkey\_generate("0102030405060708090a0b0c0d0e0f10111213141516171819","cdcdcdcdcdcdcdcdcdcdcdcdcdcdcdcdcdcdcdcdcdcdcdcdcdcdcdcdcdcdcdcdcdcdcdcdcdcdcdcdcdcdcdcdcdcdcdcdcdcd")  
**length of key 25**

**length of message 100**

**secret key generated: 10c4c1e3426a1fd0cb66e9aefcb57df89621a0a10e746f80ba6e53d0a73278bc**

#test case 5

secret\_key = secretkey\_generate("0c0c0c0c0c0c0c0c0c0c0c0c0c0c0c0c0c0c0c0c","Test With Truncation")

**length of key 20**

**length of message 20**

**secret key generated: a3b6167473100ee06e0c796c2955552bfa6f7c0a6a8aef8b93f860aab0cd20c5**

#test case 6

secret\_key = secretkey\_generate("aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa","Test Using Larger Than Block-Size Key - Hash Key First")

**length of key 131**

**length of message 54**

**secret key generated: 60e431591ee0b67f0d8a26aacbf5b77f8e0bc6213728c5140546040f0ee37f54**

#test case 7

secret\_key = secretkey\_generate("aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa","This is a test using a larger than block-size key and a larger than block-size data. The key needs to be hashed before being used by the HMAC algorithm.");

**length of key 131**

**length of message 152**

**secret key generated: 9b09ffa71b942fcb27635fbcd5b0e944bfdc63644f0713938a7f51535c3a35e2**

Reference Websites for validating test cases

<https://www.liavaag.org/English/SHA-Generator/HMAC/>

<https://cryptii.com/pipes/hmac>  
A screenshot of a cell phone

Description automatically generated

Test case proving the outcomes in RFC are correct:

<https://stackoverflow.com/questions/16232040/testing-hmacsha256-signature>