Homework Number: HW04

Name: Andrew Wu

ECN Login: wu1795

Due Date: 2/13/24

Encrypted Output:



Decrypted Output:

Newly re-signed McLaren driver Lando Norris is confident that the team will be in the mix for race victories in 2024, but the Briton feels he may have to wait a little longer for a championship challenge. McLaren caught the eye last season by going from struggling to score points to regularly fighting for podiums, with highly effective upgrades being implemented following a technical reshuffle. Norris came close to scoring McLaren's first Grand Prix win since 2021 on several occasions, taking six P2 finishes, while team mate Oscar Piastri managed to triumph in the Qatar Sprint Race.

In my AES.py file, I have split the file based on the functions being used for encryption or decryption, with the adapted lecture code at the bottom. For encryption, I read the input by blocks of 128 bits and pad them when necessary. I then slice the blocks to fill the initial state array, and XORing the initial state array with the first four words of the key schedule, derived from using the gen\_key\_schedule\_256 function. I then use a for loop to push the state array through the functions subbytes, shiftrows, mixcolumns, and addroundkey, omitting the mixcolumns function on the 14th/last round. For the subbytes function, I push each element of statearray through the gen\_subbytes\_table function that finds the multiplicative inverse of each element, XORs the element with 4 shifted versions of itself, and the special constant. For the shiftrows function, I shift the second, third, and fourth rows by one, two, and three bytes to the left, respectively. For the mixcolumns function, I multiply each element in statearray by 2, XOR them with 3 times the next element and the remaining two elements columnwise. For the addroundkey function. I slice the corresponding words from the key schedule and XOR them with statearray. I then flatten the resulting statearray into a singular bitvector and write the output in hex. For decryption, I open, read, and create blocks of one byte from the input encrypted file, and create the same key schedule as encryption. I then slice the input to get elements to fill the initial state array, just like in encryption, and conduct the initial XOR with the last four words of the key schedule. I then do the reverse round order conducted in encryption, pushing the state array through invshiftrows, invsubbytes, addroundkey, and invmixcolumns, with the last round (round 1) omitting the invmixcolumns functions. The addroundkey function is the same as in encryption, with the invshiftrows shifting right instead of left, invsubbytes doing the reverse calculation as subbytes, and invmixcolumns doing the reverse mixing as mixcolumns. I then flatten the corresponding state array and write the output in ASCII.