ECE 404 HW 7

Nimal Padmanabhan

March 9, 2023

1 Programming Problem

For this programming problem, we have implemented the Secure Hash Algorithm for the 512-bit version (SHA-512) for encrypting a small text file using Avi Kak's SHA-256 implementation in Python. The input text file contains the following: **The phony war is over and it will soon be time to discover who's hot and who's not on the 2023 Formula 1 grid. Red Bull ended last season in dominant shape, winning all bar one of the grand prix in the second half of the 22-round championship. Because of that - and their 2021 budget cap breach - they have less time to spend on developing their RB19. Will that allow Ferrari and Mercedes to reduce their advantage?** Using the SHA-512 algorithm and comparing the output with Python's hashlib library: the hash of this message is the following: **5b11ec306b005aa885c0fb9c7c286caf9e261538495944b9550 8698aeea61f552ad85c 564210088bd3f25669c89da2fdd79ee8024f1eb8d1c0bffe948637191**.

Next, we will go over the steps of doing hashing a message using the SHA-512 algorithm. The first step is padding the message from the input file to make the length of the BitVector an integral multiple of the blocksize. For SHA-512, the blocksize is 1024 bits, which means the last 128 bits of the last block should have a value that contains the length of the message. The second step is to make a message schedule needed for processing the 1024-bit input message block. This means that the message schedule contains 80 64-bit words. Next, we apply round-based processing for each of the 80 words, which consists of first storing the hash values calculated from the previous message block using 8 temporary variables. In each round, we permute those values stored in the 8 variables and mix in the message with two of the variables. Ultimately, we update the hash values calculated for each of the four variables, which will create our SHA-512 hash of the message.