Manual for Scan Attack

Steps:

1. In order to perform the scan-attack we need to extract the intermediate results i.e. the ciphertext after the first round. Relevant changes have to be made in the given aes.c code for it to run for only one round.

```
void Cipher(state_t* state, uint8_t RoundKey[176])
{

  uint8_t round = 0;

  // Add the First round key to the state before starting the rounds.
  AddRoundKey(0, state, RoundKey);

  // There will be Nr rounds.
  // The first Nr-1 rounds are identical.
  // These Nr-1 rounds are executed in the loop below.
  Cipher_label8:for (round = 1; round < 2; ++round)
  {

    SubBytes(state);
    ShiftRows(state);
    MixColumns(state);
    AddRoundKey(round, state, RoundKey);
  }
//  printf("\n HERE WE GO: \n");
//  int i,j;
//  for(i=0;i<4;i++)
//  {for (j=0;j<4;j++)
//  printf(" %x",state[i][j]);
//
//  }
//  printf("\n EASY PEAZY LEMON SQUEEZY \n");
//  // The last round is given below.
  // The MixColumns function is not here in the last round.
//  SubBytes(state);
//  ShiftRows(state);
//  AddRoundKey(Nr, state, RoundKey);
}
```

2. Once the above changes are run the C- simulation and verify first round results with Scan-attack paper.
3. Once second step is done replace the cipher function with a new function called Cipher2 (from GitHub) as shown below.
   https://github.com/deepraj88/EL9453_Hardware_Security_and_Trust/tree/master/Lab2

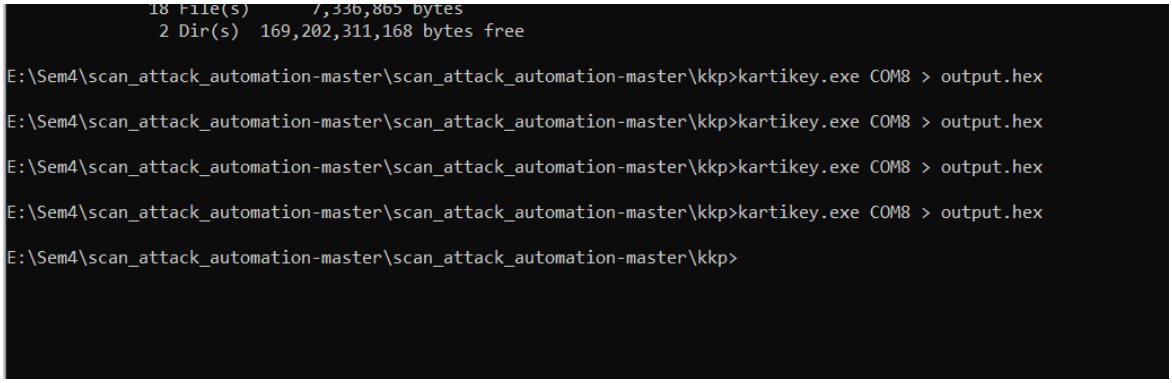
```
void Cipher2(state_t* state, uint8_t RoundKey[240])
{
  uint8_t tpp,jpp;
  uint8_t ggf[17];
  uint8_t round = 0, i = 0, j = 0;
  char temp_state[33];
  //printf("in cipher2\n");
  int quotient, remainder;
  FILE *fp, *fpp;
  uint8_t tepp;
  int jq = 0;
  fp = fopen("input.hex","w");
  for(i=0,j=0;i<192;i++)
  {
  fclose(fp);
  system("Scan.exe COM4 > output.hex");
  quotient = 0;
  remainder = 0;
  //printf("exe executed.\n");
  fpp = fopen("output.hex", "r");
  if ( fpp == NULL )
    {
    else
    {
    fclose(fpp);
}
#else
```

4. For scan attack different input plaintext must be given to generate different intermediate results and XORing the successive intermediate to check the number of ones in a byte and if they are either 9,12,23 or 24 then replace those bytes with certain values that is given in the scan attack paper given by the Professor.
5. Now download the **Scan.exe** from the github which is a wrapper for extracting the intermediate results from the FPGA.
6. Since this Scan.exe file uses java libraries make sure to install the java libraries and set environment variable before running it (jdk-13.0.2 is preferred). Below is the link from where the jdk libraries can be downloaded.

   https://www.oracle.com/java/technologies/javase-jdk13-downloads.html

7. Create input.hex file containing the input vectors(plaintext) and keep it the same location as the Scan.exe file.
8. Now burn the .bit file on the FPGA and the run the .exe file from the command prompt in the windows in the below format. Output.hex is where your output from the FPGA is stored.

   Scan.exe COM<port number specific to your laptop> output.hex



9. Once you get the output.hex file apply step 3 and then write a C code to zero in on the actual key.

IF you are facing difficulty while executing code or getting any errors like Port cannot be opened then ask for **HELP!!!!**