DES Lec #3, AES Lec #1

Monday, October 5, 2020 11:43 AM

Agenda

1) Complete description of DES key schedule
2) Walk through DES code online
3) Start AES

Key Schedule[1]

Input Key is 56 bits, but is presented as 64 bits with 8 checksum bits.

Let's label the key bits k_1 through k_{64} and of these bits, the checksum bits are k_8, k_{16}, k_{24}, k_{32}, k_{40}, k_{48}, k_{56}, k_{64}.

Checksum is ODD PARITY for the row.

1011 0011, that last bit, k_8 has to be 1 so that there are an odd # of 1s on the row.
0111 1100,
1100 0001,
0000 0010,
0010 1111,
1111 1110,
0101 1110,
1101 1111

In most books they would list the key in HEX:

b3 7c c1 02 2f fe 5e df

Algorithm

Take the initial key and split it in half, calling the left half C_0 and the right half D_0. Note that our bit numbers will range all the way to 63 since we keep the bit numbers from the original numbering.

1) PC-1(C_0D_0)

1) For 16 rounds, do this:
   for i = 1 to 16:
      $C_i = LS_i(C_i)$
      $D_i = LS_i(D_i)$
      $K_i = PC\text{-}2(C_iD_i)$

Left-Shift is a cyclic left shift of the buffer, by either 1 or 2 bits. There is a table that tells you whether to do 1 bit or 2.

After PC-1 Step - bits we grab from original key
57 49 41 33 25 17 9 1 58 50 42 34 26 18 10 2 59 51 43 35 27 19 11 3 60 52 44 36

63 55 47 39 31 23 15 7 62 54 46 38 30 22 14 6 61 53 45 37 29 21 13 5 28 20 12 4

Calculate C1 and D1 by cyclic left shift on both buffers round 1 - it's 1 bit

49 41 33 25 17 9 1 58 50 42 34 26 18 10 2 59 51 43 35 27 19 11 3 60 52 44 36 57

55 47 39 31 23 15 7 62 54 46 38 30 22 14 6 61 53 45 37 29 21 13 5 28 20 12 4 63

We will apply PC-2 to this entire buffer above. Here is PC-2:

14 17 11 24 1 5 3 28 15 6 21 10 23 19 12 4 26 8 16 7 27 20 13 2
41 52 31 37 47 55 30 40 51 45 33 48 44 49 39 56 34 53 46 42 50 36 29 32

The round 1 key, starts with the 10th bit from the original key, followed by the 51st bit from the original key, followed by the 34th bit of the original key, followed by the 60th bit of the original key.

If I wanted to speed up my encryption decryption, I would pre-calculate all 16 round keys as permutation matrices of which bits to select from the original key. The Round 1 Key matrix would start: 10, 51, 34, 60, …

Before I know the key, we can calculate which bits to grab from the

original key in forming each round key.

You can have 16 tables of size 48, each of which tell you which bits to grab from the original key to form the 16 round keys.

(These tables are printed in Stinson, from the extra book list…)

Code Walk Through

Left Shift Function

			s			e		
0	1	0	0	0	1	0	1	0

create temp for my example for size 4, e-s+1, let us do a cyclic left shift of 2, so yellow is my starting index. That is start + numbits. But we want to wrap around… start + (numbits+i)%size

temp

1	0	0	0

Then we copy back into the appropriate section:

			s			e		
0	1	0	1	0	0	0	1	0

whole is offset by start…

Get started with AES

So, by the mid 1990s after DES had be around 20 years or so, computers had gotten a lot faster, and were getting to the point where maybe very powerful computers could try all possible keys to attach DES.

In 1998, the DES challenge was posed and utilized distributed computing to break a DES key via brute force.

Server that would farm out sets of keys to try to computers that had were IDLE. just had to be networked and donate your spare cycles.

Back then, using 1998 computing power and distribution, the key was broken in 3 months.

This time span wasn't considered to be adequately long for security purposes. It's possible that information from 3 months ago could still remain valuable and should not get into unwanted hands.

At this time, the government solicited for a new algorithm with improved security so that a brute force search was infeasible in a reasonable amount of time.

Goals for the submitted algorithms:

1) Security (should not be breakable in a reasonable amount of time)
2) Simplicity(should be easy to understand and easy to implement in both software and hardware)

Initially 15 algorithms considered, which were paired down to 5 finalists, and ultimately, the algorithm Rijndahl (prounouced Rain Doll) won. The name comes from a concatenation of the two creators of the algorithm:
Vincent Rijmen and Joan Daemen,

From <https://en.wikipedia.org/wiki/Advanced_Encryption_Standard>

Algorithm has 3 possible versions:

V1: 128 bit key, 128 bit block
V2: 192 bit key, 192 bit block
V3: 256 bit key, 256 bit block

Idea here is that if 128 bit key becomes brute forcible in the future, we can just move to a different version.

I will just cover the 128 bit version with you.

The computing power necessary to break 128 bit key is much, much more than to break a 56 bit key.

Largely non-mathematical, but the security is based in some number theory. So I'll just teach you the very minimal amount of number theory necessary to trace through the steps of the algorithm.