# MATH 314 - Class Notes

10/24/2017

Scribe: Megan Clark

**Summary:** We went over different mode of operation for enrypting larger than 64 bits and briefly began talking about AES/SAES.

**DES:**

- Key length is too short (56 bits) to be secure today
- First solution was 2DES but a meet in the middle attack makes this insecure

**3DES:**

- 3 keys K1,K2,K3
- perform encryption by: C=(E(K3)(E(K2)(E(K1(p)))))
- what happens if we try a meet in the middle attack now? D k3(C)=E k2(E k1(p))
- no matter how we try to manipulate the equation we always have 2 keys on one side of the equation and by having two you would need $2^{112}$ entries to store which is too big to store, so since this is too big to store it is secure against a meet in the middle attack

**Actual 3DES:**

- uses 2 keys K1, K2
- Encryption: (E(K1)(D(K2)(E(K1(p)))))=C
- Decryption: (D(K1)(E(K2)(D(K1(C)))))=P
- 3DES is still used today (considered secure) but better to use newer algorithms

**How do you encrypt something larger than 64 bits?:**

- First Solution:Break into blocks of 64-bits and encrypt each block seperately, this is called Electronic code book (ECB)
- Flaw of ECB: Same plaintext always encrypts to the same ciphertext
- use frequency analysis on entire blocks to perform an attack
- to solve this problem use different modes of operation

**Modes of Operation:**

- ECB is one mode of operation
- Cipher Block Chaining(CBC)

- Cipher feed back (CFB)
- Output-Feedback (OFB)
- Counter(CTR)

**CBC:**

- first break the plaintext into 64 but blocks: p1,p2,....
- Pick an initial C0 in cleartext (means without encryption)
- Method to compute blocks of cipher text:
- Ci=E(Pi + Ci-1)
- Pi=D(Ci)+Ci-1-Pi
- How does this solve the problem of ECB?
  - Suppose we send the same message over and over again P1=P2=P3
  - then C1= E(P1+C0), C2=E(P2+C1)
  - Even though we have started off with p1 and p2 equal they are being XOR with different random bits so they are going to be completely different
  - So C1 != C2 != C3, we use the previous ciphertext as a one-time pad to futher scramble the plaintext
- CBC Issues
- has issues with propogation, one error transmission confuses the decryption of later blocks

**CFB:**

- Break plaintext into 8-bit blocks (performing encryption one byte at a time)
- Fix X1= random 64 bit string send in clear text
- Compute Oi=L8(E(Xi)) L8= leftmost 8 bits
- Ci=Oi+Pi
- Xi+1= R56(Xi) —— Ci R56= rightmost 56 bits
- This is a one-time pad where encryption function is used to get the bits used on the pad

**OFB:**

- same set up as cipher feedback creat initial x1 as before
- compute:
  - Oi=L8(E(Xi))
  - Ci=Oi+Pi

  - Xi+1=R56(Xi) —— Oi
- The only difference from ecb is for xi+1 we append it with Oi instead of Ci
- The benefit of OFB is that the Oi's cam be precomputed before the plaintext is known (it can help run faster)
- CFB is a little more secrue but OFB is faster

**CTR:**

- Xi=i(mod $2^{64}$)—-in binary
- Oi=L8(E(Xi))
- Ci=Oi+Pi
- Same benefits as OFB but easier to compute

**AES:** Advanced Encryption System

- By the mid 90's it was clear that DES needed to be replaced (main flaw: key length not long enough), now they could have chosen to just make the key longer but decided to undergo to fix all flaws
- NIST: called fir proposals for a replacement for DES using new cryptology advancements
- Flaws wanted corrected for DES:
  - Longer key length
  - 16 rounds slower than ideal(run faster)
  - use math that is more clearly generated how the diffusion/confusion occur
  - Mysterious SBOXES
- 5 different proposals were given NIST chose one called Rijndael(Raindoll)
- This became the new AES

**AES:**

- not fiestal system (lets us do fewer rounds)
- 4 key steps to a round of SAES
  - 1. Add Round
  - 2. Substitue
  - 3. Shift Rows
  - 4. Mix Columns