CS 346 Class Notes

Mark Lindberg

Mar 23, 2016

Last Time:
Chapter 6: Practical constructions.
This Time:
Continued!
Substitution-Permutation Networks: (SPNs). (AES block cipher is an example of this.)
Each "round" has 3 OPS:

1. Key mixing. $x \to x \oplus k$, where k is the sub-key for the round i.
2. Substitution. Break up x into pieces and apply "s-boxes". (Each s-box sounds like a permutation.)
3. Permutation. Permute the bits of x post-substitution by step 2.

Given the key, k (And hence, $\forall i$, k_i), then each action of each operation, and therefore the whole process, is invertible. Permutations can be inverted, because they are one to one. With the sub-key, we can "undo" the key mixing as well.

Remark: The last round of all 3 OPS should be followed by an additional key mixing step, else the substitution and permutation steps are completely reversible, and therefore pointless.

Feistel Networks: (We'll focus on the "balanced" case.)

We take our n-bit input and divide it into two halves. (Balance means we can assume clean division.)

Call these halves L_0 and R_0, both of which are $\frac{n}{2}$-bits.

Let f_i be the round function for round i.

For round i, compute $R_i = L_{i-1} \oplus f_i(R_{i-1})$, and $L_i = R_{i-1}$. (Run the computation and swap them.)

Here, we *can* let f_i be non-invertible.

To invert the whole thing, $R_{i-1} = L_i$, and $L_{i-1} = R_i \oplus f_i(R_{i-1})$.

DES uses a 16-round balanced Feistel Network.

- The round function is the same in each round (Except for round sub-key). The function is sometimes called the DES "mangler" function. It has an SPN-like structure.

In DES, the block side is 64 bits. The key size is 56 bits. (This is short, one can brute-force it today because of advances in computing.) [There is a variation called triple-DES which addresses this.]

Our round function takes a $64/2 = 32$-bit input (And produces a 32-bit output, of course).

The sub-key in each round is 48-bits, formed by a permutation of a subset of the key's bites.

There is a trivial expansion step during which half of the bits of the input are duplicated to get a 48-bit result.

This is XORed with the sub key. This gives the key-mixing step.

Next, We break the mixed key into 8x 6-bit chunks in the obvious way.

The ith chunk goes to s-box s_i which gives us a "permutation" which produces 4 bits. This is the analogue of step 2. It's a little different in that the value shrinks. We get a 32-bit value from the s-boxes.

There is then a 32-bit permutation which is applied to the 32 bits, giving us a 32-bit output.

Mangler function!

Increasing the key-size of a block cipher. (Using DES as an example.)

Idea 1: "double encryption", $F_{k_2}(F_{k_1}(x)) = y$, but this still admits an attack using roughly 2^{56} evaluations.

Idea 2: "triple encryption", $F_{k_1}(F_{k_2}(F_{k_1}(x))) = y$. This one needs 2^{112} evaluations, so it's good and secure.

Hash functions in practice!

SHA, SHA-2, and others, are based on a Davies-Meyer construction (Function : $2n \to n$ plus the Merkle-Damgård function.

Davies-Meyer

- Uses a block cipher to get a compression function.

Can be justified in "ideal-cipher" model. That is, it can be proved collision resistant in the "ideal-cipher" model.

SHA-3.

We don't need to bound the running time of $\mathcal{A}$. Just need to bound the number of block cipher evaluations.