MATH 587/CSCE 557 - SUMMARY OF CLASS, 4/12/07

We discussed Feistel ciphers. Let $\{0,1\}^m$ denote the set of bit strings of length m. If x, y are elements of $\{0,1\}^m$ and f any function from $\{0,1\}^m$ to itself, then we let $x' = y, y' = x + f(y)$, where addition is bitwise (mod 2). The neat thing is that this is invertible whatever f is. The idea is to do several Feistel rounds with different functions $f_1, f_2, ...$

If f_1 and f_2 are constant functions, say $f_1(x) = x + k_1, f_2(x) = x + k_2$, where k_1, k_2 are elements of $\{0,1\}^m$, the same whatever x is, then the two Feistel rounds lead to $x'' = x + k_1, y'' = y + k_2$, which is just Vigenere with keyword length $2m$. We have cryptanalysis methods and also since composing Vigenere ciphers yields a Vigenere cipher, further rounds add no further security. If we use a linear f, i.e. $f(x) = Mx$, where M is a matrix with binary entries, then we just get a Hill cipher and again cryptanalytic techniques are known.

We began discussing DES (Data Encryption Standard). This is still widely used, has historical importance, and is a well-designed cipher. To describe it, we need Boolean functions. These are functions from $\{0,1\}^m$ to $\{0,1\}^n$ for some m, n (called m-in, n-out). If $m = n$ and the function just permutes the bits, we call it a permutation. If $m < n$ and the function maps different strings to different strings (i.e. is 1-1), then we call it an expansion.

To summarize, DES takes a 56-bit key k and a 64-bit plaintext, applies a specific permutation σ to the plaintext, applies 16 Feistel rounds (so $m = 32$) using keys $k_1, ..., k_{16}$, and then applies the inverse of σ to get the 64-bit ciphertext.

How to obtain $k_1, ..., k_{16}$. The 56-bit key is expanded to a 64-bit key with the 8th bit the sum of the first 7 bits, the 16th bit the sum of the next 7 bits, etc. A specific permutation is applied to the 56 bits and a clocking schedule is applied that cyclically permutes the key by one or two before each round. The 48-bit subkeys $k_1, ..., k_{16}$ are then obtained by applying specific 56-in 48-out Boolean functions.

These keys are then used to create the Feistel rounds. Consider the output of the previous Feistel round as x, y with each of x, y 32-bit. Expand y to y^* using a specific 32-in 48-out Boolean function. Add the subkey k_i to y^* to make z. Write $z = z_1...z_8$, where each z_i is 6-bit. Much of the security for DES comes from the S-boxes which are specific 6-in 4-out Boolean functions. We compute $w = S_1(z_1)...S_8(z_8)$, which has 32 bits. Then a specific permutation of order 2 is applied to w to make the output. This is how $f_i(y)$ is produced for the ith Feistel round.

The key schedule is linear so security comes from the S-boxes. There are about 10^{77} possible 6-in 4-out Boolean functions. Design criteria include: no output bit being too close to a linear or affine function of the input bits. Using Hamming distance d, $d(x, x') = 1$ should imply that $d(S(x), S(x')) \geq 2$. Finally, if we fix c in $\{0,1\}^6$ and consider all x, x' such that $x - x' = c$, there are 2^6 of them. Out of this, a given $S(x) - S(x')$ should arise from no more than 8 pairs. This defends

against differential cryptanalysis, which apparently NSA knew about at the time (but others didn't) and helped build into the S-boxes.