1/28/2004

Notes for Lecture 4

4 Block Ciphers

4.1 Definition

A block cipher is an efficiently computable keyed permutation $e_k : \{0,1\}^s \to \{0,1\}^s$. If the key k is known, then the inverse e_k^{-1} is also efficiently computable. The value of s is called the *block size*.

4.2 DES

In the 70s, banks began to network their computers. Some form of cryptography was needed to protect transactions between banks and other commercial transactions. The body that is today known as the U.S. National Institute of Standards and Technology issued a call for encryption algorithms. The resulting algorithm was based on a design that originated at IBM and became the DES, or Data Encryption Standard.

We will now poke briefly into the guts of the Data Encryption Standard. Our glimpse should convince us that we want to stay on this side of the abstraction barrier.

DES is a 64-bit block cipher that takes a 64-bit key, 8 bits of which are parity checks, yielding 56 bits of "real" key. An input block is broken into a left block and a right block, both of 32 bits. The 56 bits of key are then expanded using a *key schedule* into 48-bit *subkeys*, one for each of 16 *rounds* of DES. In each round, the left block is passed through a function f and combined with the round subkey. The result is then XORd with the right block, and then the blocks are swapped.

One of the remarkable things about DES is that even though the f function in each round is complicated, non-linear, and *not* a permutation, the DES function can be inverted. To invert, we "run DES backwards." The left half of the ciphertext is run through the F function with the key of the last round, XOR'd with the right half of the ciphertext, and then the halves are swapped. This split-in-halves-and-swap structure is called a *feistel structure*, after Horst Feistel, who was on the team at IBM that developed DES.

4.3 AES

DES was the subject of intense analysis for many years. In the end, however, what limited DES was the size of its key. Given today's technology, 56 bits is too short. The Electronic Frontier Foundation, working with Cryptography Research, built a DES cracking machine in 1998 that could brute-force DES keys within a week, for a cost of about $US\$250,000$. Something new was needed.

NIST held a competition to select an *Advanced Encryption Standard* or AES. After much work, in 2000 a winner was announced. The winning algorithm was called *Rijndael*, and it was invented by Joan Daemen and Vincent Rijmen.

AES has multiple parameter settings, but we will briefly consider the case of a 128-bit key and 128-bit block size.

4 Notions of Security for Block Ciphers

Now that we have a basic definition and some examples, let's look at notions of security for block ciphers. The following are some attack notions used by practicioners. You will sometimes see papers

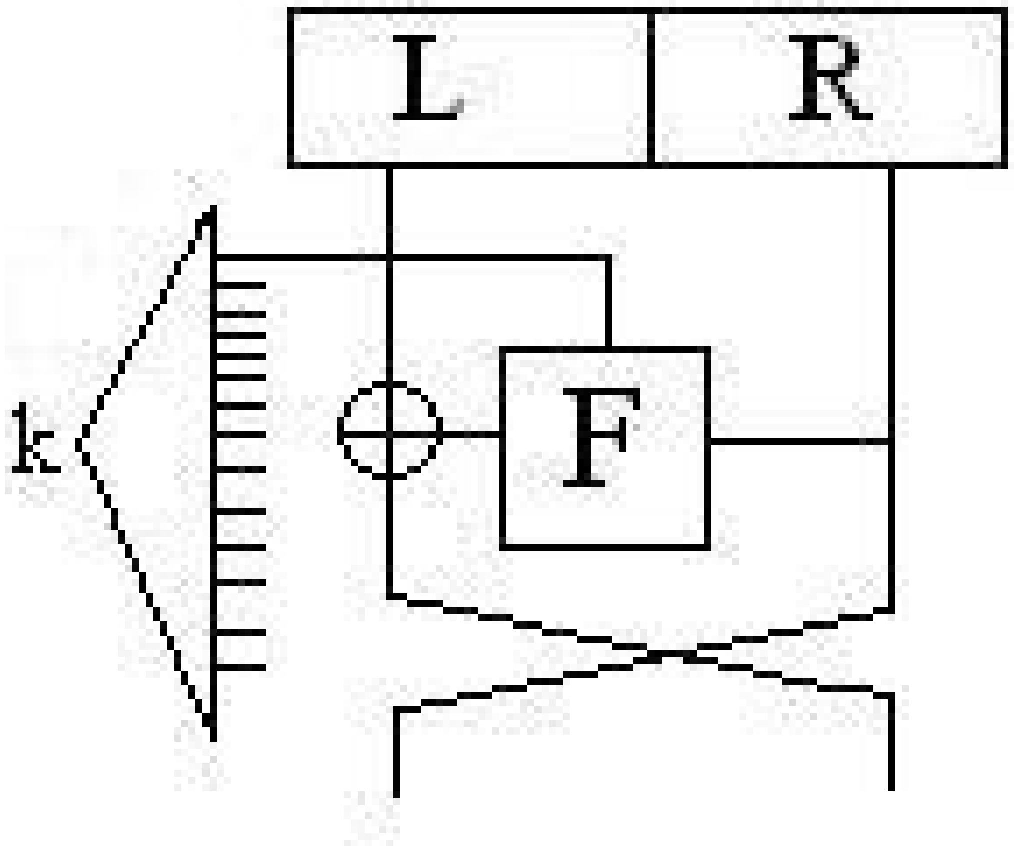


Figure 1: One Round of DES

quantify their attacks in terms of how many plaintexts or ciphertexts are required for each of these attacks.

4.1 Ciphertext-only

In a *ciphertext-only* attack, the adversary is given ciphertext and nothing else. This is the weakest form of attack.

4.2 Known-plaintext

In a *known-plaintext* attack, the adversary is given many plaintexts and their corresponding ciphertexts. From these, the adversary may be able to obtain the key or a rule for decrypting future messages.

4.3 Chosen-plaintext

A *chosen-plaintext* attack goes further; the adversary is given the power to choose plaintexts of its choice and see their encryptions.

4.4 Chosen-ciphertext

The *chosen-ciphertext* attack gives an adversary the power to ask for the *decryptions* of arbitrary strings. Unlike the other attacks, chosen-ciphertext attacks allow the adversary to probe e_k^{-1} instead of e_k.

4 Modes of Operation

A block cipher is a permutation on s bits. Usually we want to encrypt a message that is not exactly s bits; for smaller messages, we can pad with 0 to make the message exactly s bits. There are several methods for using a block cipher to encrypt larger messages, which are collectively known as *modes of operation*.

4.1 ECB

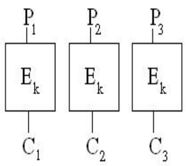
The first method we consider is *Electronic Code Book*, or ECB mode. In ECB mode, plaintext is broken into s-bit blocks, and then each block is encrypted independently, as in the following diagram.

This method, to use a technical term, *blows chunks*. What happens if $P_i = P_j$? In that case, $C_i = C_j$, and an adversary can tell that a plaintext block has been repeated. This is undesirable; what if the message is a string of stock orders, and blocks encode messages such as "buy IBM" and "sell IBM" ?

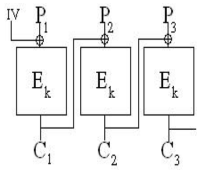
4.2 CBC

The next mode we consider is called *cipher block chaining*, or CBC mode. As in ECB mode, we will consider the message as broken into s-bit chunks. In CBC mode, each plaintext block is XOR'd with the previous ciphertext block before encryption. In symbols, $C_i = e_k(P_i \oplus C_{i-1})$.

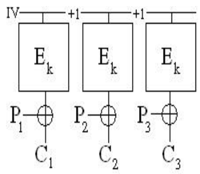
We start the process by using a "dummy" ciphertext block called an *initialization vector* or IV. The IV need not be secret.



(a) ECB



(b) CBC



(c) CTR

Figure 2: Modes of Operation

4.3 CTR

In CTR mode, an incrementing counter is encrypted and the output used as a "keystream" to XOR against the plaintext. In symbols, we have $C_i = e_k(ctr_i) \oplus P_i$, where $ctr_i = ctr_{i-1} + 1$ and the value of ctr_0 is given by an IV.

The IV may be the result of state, may be picked randomly, or may be set to 0 each time. Using a block cipher in CTR mode is a way to convert a block cipher to a kind of pseudo-random generator.

4 PRFs and PRPs

We now turn to two notions that capture what we might want from a block cipher.

4.1 Pseudorandom Functions (PRFs)

For our first notion, we might say that we want the block cipher's output to "look random." Put another way, no adversary, faced with the task of distinguishing between a good block cipher and a truly random function, should be able to tell which is which. Formalizing this gives us the following definition. We say that e_k is a (t, ϵ)-PRF if for all probabilistic polynomial time adversaries A running in time at most t, we have

$$|\Pr_k[A^{e_k} = 1] - \Pr_{f \leftarrow_R \mathcal{F}}[A^f = 1]| \leq \epsilon$$

where $\mathcal{F}$ is the set of all functions from $\{0,1\}^s \rightarrow \{0,1\}^s$, and $\leftarrow_R$ means "drawn uniformly at random". As a reminder, the notation A^{e_k} means that the adversary A is given the function e_k as an *oracle*.

4.2 Pseudorandom Permutations (PRPs)

In the above definition, we drew uniformly at random from the set of *all functions* from k bits to k bits and asked that our pseudorandom function be indistinguishable from such a function. In the case of block ciphers, however, we know that we are dealing with permutations. It can be shown that "most" functions are not permutations, so in some sense our block cipher is "far" from being a random function. We can address this by defining a notion of *pseudorandom permutation* or PRP.

We say e_k is a (t, ϵ)-PRP if for all adversaries A running in time at most t, we have

$$|\Pr_k[A^{e_k} = 1] - \Pr_{f \leftarrow_R \mathcal{P}}[A^f = 1]| \leq \epsilon$$

but here, $\mathcal{P}$ is the set of all permutations on s bits.

4.3 Examples of PRFs and PRPs ?

Now that we have defined PRFs and PRPs, let us look for some examples. We can begin with some simple attempts. In the following, k is the secret key:

1. Is $e_k(x) = x$ a PRF or PRP ? We certainly hope not, and in fact it's easy to disqualify. While e_k always maps x to itself, this event happens for a random function or a random permutation with much lower probability.

2. Is $e_k(x) = x \oplus k$ a PRF or PRP ? No, since $e_k(x) \oplus e_k(y) = x \oplus k \oplus y \oplus k = x \oplus y$, which is unlikely with a random function or permutation.

3. Is $e_M(x) = M \cdot x$, where M is a secret square matrix, a PRF or PRP? Certainly not, since e_M is linear: $e_M(x) + e_M(y) = e_M(x + y)$, and a random function or random permutation is linear only with small probability.

It looks like constructing a *PRF* or *PRP* is going to be difficult. We will come back to this question in future lectures and offer some more candidates.