COMPUTER & NETWORK SECURITY

Lecture 4:

Cyphers II

PSEUDORANDOM NUMBER GENERATORS

Sources of random numbers are desirable in many occasions:

Session Keys, Deck Shuffling, Challenges, Nonces, ...

Unfortunately, truly random sources are not easy to come by...

Thermal noise of electrical circuits, timing of Geiger counter clicks

Really hard to come by...

U.S. Patent 5,732,138:

"Method for seeding a pseudo-random number generator with a cryptographic hash of a digitization of a chaotic system."

(called **Lavarand** by **Silicon Graphics**: take a guess what it is)

Instead, most applications need to make do with a pseudorandom number generator (**PRNG**)

PSEUDORANDOM NUMBER GENERATORS

Desirable properties of PRNGs include:

- Repeatability
- Statistical randomness
- Long period / cycle
- Insensitive to seeds

PRNGs are often broken by:

- Statistical tests that find patterns or biases in the output sequence
- Inferring the state of the internal registers from the output sequence

PRNGs are usually **critically important parts of the system**,
and often **a single point of failure**

LINEAR CONGRUENTIAL GENERATORS

$$x_{n+1} = (ax_n + b) \bmod c$$

Unix rand() function

a, **b**, **c** are constants

Period of generators is less than **c**

Cannot be used for security – easily predictable

Only need two consecutive values to reconstruct the internal state

Used by an Internet casino who were so sure of their code, they published their algorithms! With expected results...

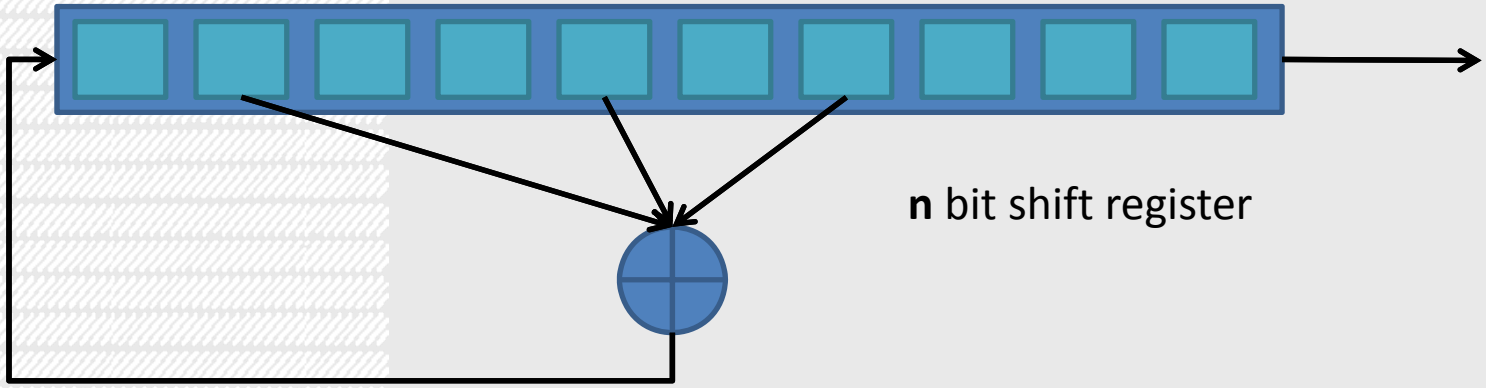
Moral of the story: don't use it

LINEAR FEEDBACK SHIFT REGISTERS (LFSR)

Seed is the initial value of the shift register

Feedback network based on polynomials over finite fields

Easy and fast in hardware (1 bit per clock)



Core Problem: Tap configuration can be determined from **2n** output bits

THE RC4 STREAM CYPHER

Wide applications in cryptography: over 50% of all SSL traffic and core to WEP

Based on permutations of a 256 byte array: the seed is the initial array value
RC4's key scheduling algorithm has known problems (WEP weakness)


```
i = j = 0;
while(1) {
        i = i + 1 (mod 256);
        j = j + s[i] (mod 256);
        swap (s[i], s[j]);
        t = s[i] + s[j] (mod 256);
        output s[t];
}
```

OTHER PRNGS

ANSI X9.17

Based on 3DES

DSA PRNG

Based on SHA or DES

RSAREF PRNG

Based on MD5 hashing and addition modulo 2^{128}

USING PRNGS

1. Be extremely careful with PRNG seeds!
2. Hash PRNG inputs with a timestamp or counter
3. Reseed the PRNG occasionally
4. Use a hash function to protect PRNG outputs if PRNG is suspect

STREAM CYPHERS

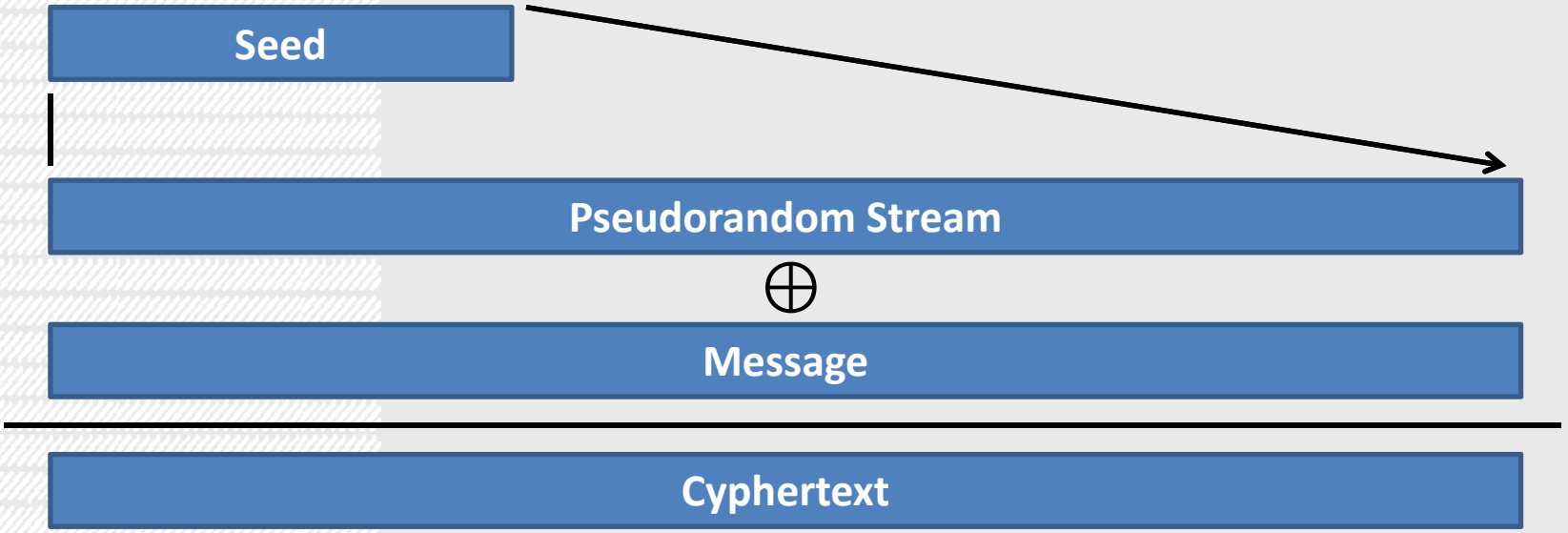
In a OTP, the secret key **k** is the random **n**-bit stream

Stream cyphers replace this random stream with a pseudorandom stream

The secret key is the seed used to generate the pseudorandom stream

$$E(m, seed) = m \oplus RNG(seed)$$

$$D(c, seed) = c \oplus RNG(seed)$$



SECURITY OF STREAM CYPHERS

Trade-off: excellent secrecy for ease of implementation / use

The security of the cypher is dependent on the security of the pseudorandom number generator:

it must be computationally hard to find the seed or the sequence

Since the PRNG is deterministic, the **seed must only be used for one session**

Stream cyphers are much faster than block cyphers

To avoid re-using the same seed, we can encrypt it using strong crypto and append it to the cyphertext to inform the other party:

$$E(m, k) = DES(seed, k) \,||\, m \oplus RNG(seed)$$

DATA ENCRYPTION STANDARD (DES)

Block cypher (64 bit blocks using a 56 bit key)

16-round Feistel network:

Feistel network is a particular construction that's reversible

$c = DES_k(m)$ $m = DES_k(c)$

(note: key schedule is reversed – you'll understand what we mean soon)

Operates in many different modes

It was the world's most heavily used and analysed cypher

We still don't understand it properly 30 years later

The NSA knew more than we do now over 20 years ago

HISTORY OF DES

1970s	IBM Research Team led by Feistel devises a cypher called LUCIFER (128 bit message, ciphertext and keyspace)
1973	NBS (now NIST) asks for a proposed data encryption standard
1974	IBM develops DES from LUCIFER
1975	The NSA “fixes” DES: shortens key to 56 bits (on 64 bit blocks), plays with S (substitution) boxes and adds additional permutations
1977	DES adopted and used heavily in financial transactions
1991	Biham & Shamir discover the NSA’s modifications made DES more resilient to differential cryptanalysis rather than less
1993	Michael Wiener from Nortel theorises a USD$1M machine could crack DES in 3.5 hours using off the shelf components
1997	DES cracked by brute force by Distributed.net in 96 days
1997	NIST asks for proposal for AES (advanced encryption standard)
1999	DES cracked by brute force again in 24 hours using Distributed.net and the EFF USD$250,000 Deep Crack machine
2000	Rijndael accepted as new AES standard (128/192/256 bit keyspace, 128 bit blocks)

THE SCARY PART: THE NSA

NSA modifications were believed to be adding a back door to the standard

It was almost 20 years later in 1990 that academia independently rediscovered and openly introduced the field of differential cryptanalysis which had been discovered by IBM in 1974 and gagged since then by the NSA

Bruce Schneier : “It took the academic community two decades to figure out that the NSA 'tweaks' actually *improved* the security of DES”

The NSA used differential cryptanalysis to *strengthen* DES when no-one else even knew it existed

"NSA doesn't want a strong cryptosystem as a national standard, because it is afraid of not being able to read the messages. On the other hand, if NSA endorses a weak cryptographic system and is discovered, it will get a terrible black eye.“

EFF 1998

FEISTEL NETWORKS

Ladder structure

Input is split into two blocks, **L**eft and **R**ight

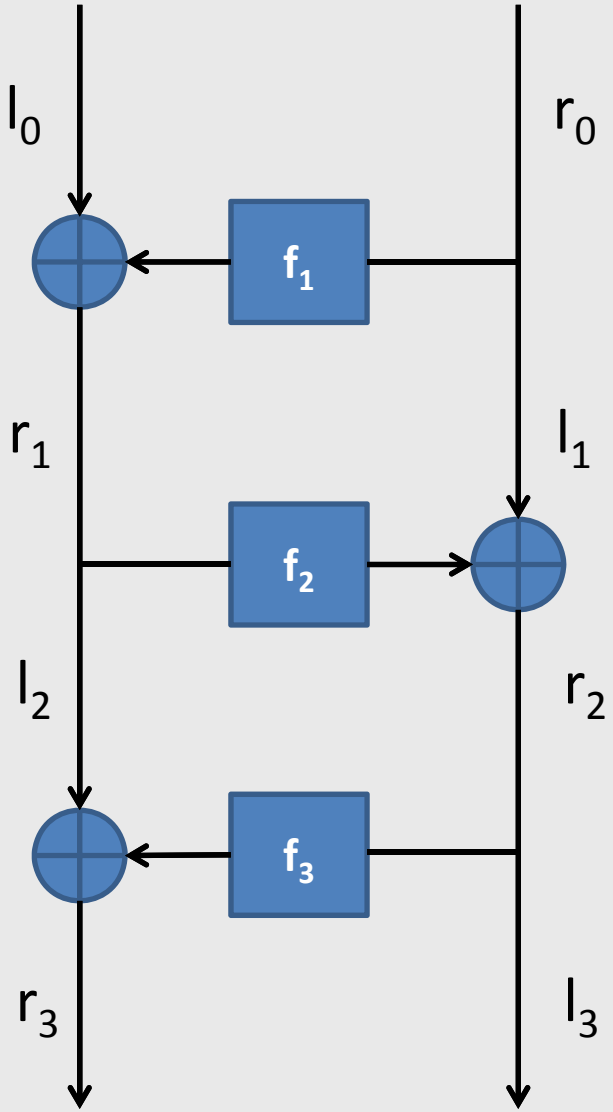
The functions $\mathbf{f_1 \ldots f_k}$ are arbitrary mappings:

$$\mathbf{f_1 \ldots f_k} : \{0, 1\}^n \rightarrow \{0,1\}^n$$

Each round:

$$l_i = r_{i-1}$$

$$r_i = l_{i-1} \oplus f_i(r_{i-1})$$



FEISTEL STRUCTURE

Express cypher as combinations of successive round functions:

$$\Psi(f_1, f_2, f_3)$$

For decryption, we use the rounds in the reverse order:

$$\Psi\text{-}1(f_1, f_2, \dots, f_{2k-1}) = \Psi(f_{2k-1}, \dots, f_2, f_1)$$

Round functions do not need to be invertible

If f_i are random functions then $\Psi()$ is indistinguishable from a random permutation under a chosen plaintext attack

This allows us to turn any one-way function into a block cypher

Now we can optimise round functions individually

DIFFUSION AND CONFUSION

Many modern symmetric cyphers are based upon two principles:

Diffusion is used to dissipate the statistical structure of the plaintext into long range statistical properties of the text

- Diffusion is achieved through repeated applications of a **permutation** function
- Sometimes known as a P-Box in cyphers
- In cypher design, we try to get the cyphertext symbol, digraph and trigraph frequencies to be as evenly distributed as possible
- Flipping one bit in the plaintext should result in a 50% probability of each bit flipping in the cyphertext output

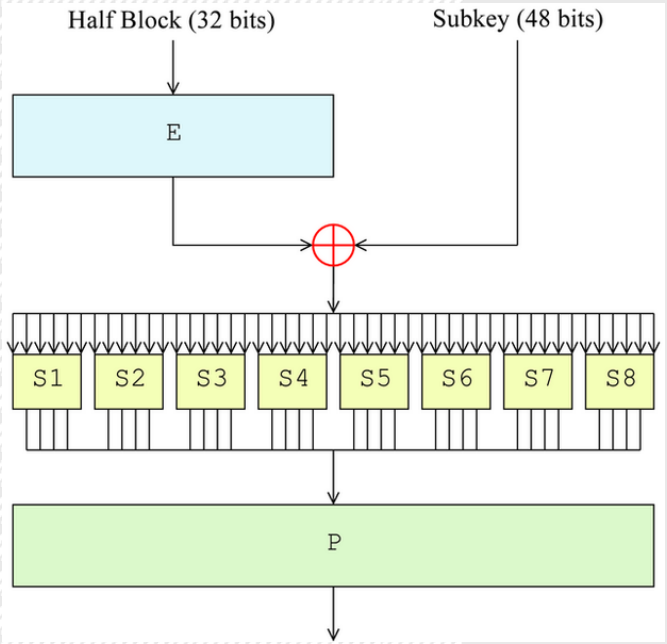
Confusion is used to make relationships between the cyphertext and the key as difficult as possible

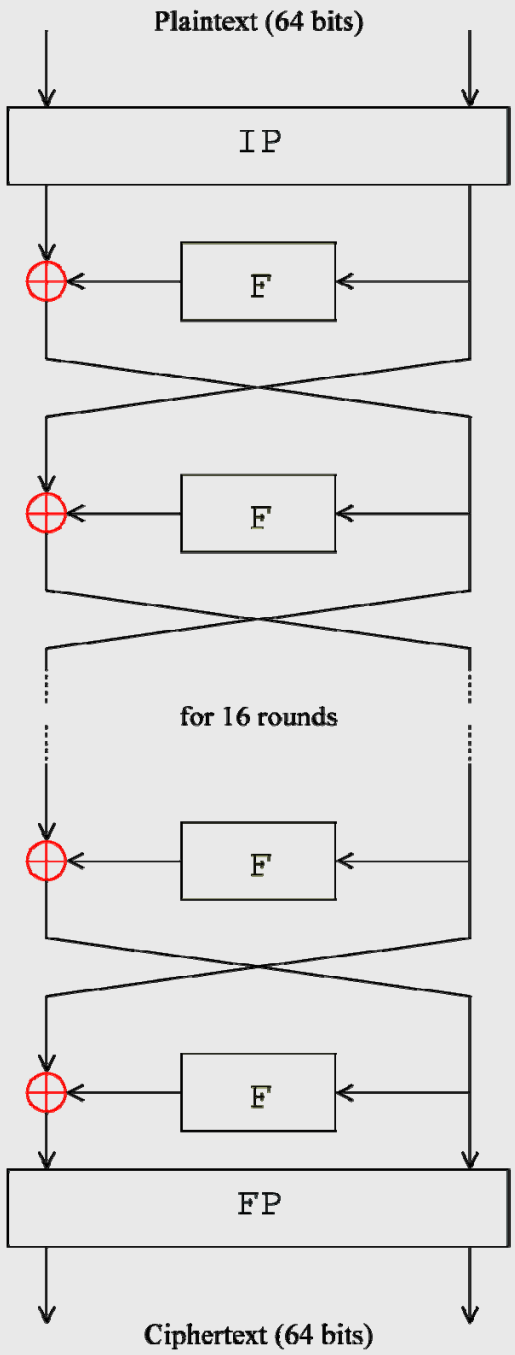
- Usually achieved through application of complex **substitution** functions
- Usually seen in the form of an n x m bit S-Box

DES STRUCTURE

The Key schedule $s_1 \ldots s_{16}$ is derived from the DES key **k**
(subkeys are 48 of the 56 bit key)

S-boxes confuse (substitution)
P-boxes diffuse (permutation)





Initial permutation to discourage software implementations (transposition)

16 rounds in total

Inverse of initial permutation

DES INTERNALS

16 round Feistel network with functions $f_1 \ldots f_{16}$ derived from the key (through the key scheduling algorithm)

DES can be defined by the following equations:

1. Split the 64 bit input into two 32 bit pieces (**L**, **R**)
   $\mathbf{M = L_0\ R_0}$
2. (repeat for 16 rounds...)
   $\mathbf{L_i = R_{i-1}}$ # 16 rounds
   $\mathbf{R_i = L_{i-1} \oplus F(R_{i-1}, k_i)}$
3. Output is defined as
   $\mathbf{C = R_{16}\ L_{16}}$ # output

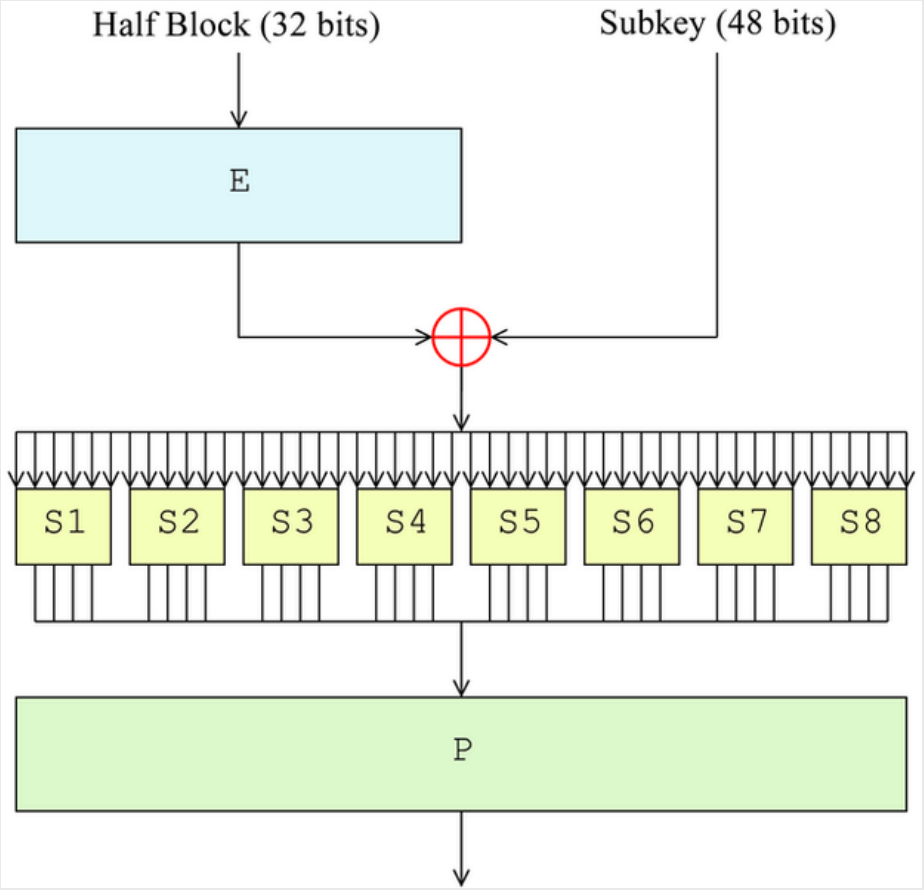
Each $\mathbf{k_i}$ is the ith subkey derived from the key **k** according to a key schedule

DES ROUND FUNCTION

The function $\mathbf{F}(\mathbf{x},\mathbf{k}_i)$: $\{0,1\}^{32} \times \{0,1\}^{48} \rightarrow \{0,1\}^{32}$

Half block is reversibly expanded
to 48 bits in the Expander (E) function

S-Box collapses groups of 6 bits into
groups of 4 bits
(i.e. convert 48 bits back to 32 bits)



AVALANCHE EFFECT IN DES

DES (and all cryptographic hash functions) is designed so a minor change in the key or the plaintext results in a dramatic change in the cyphertext

Round	Bit change in plaintext (#bits different in cyphertext)	Bit change in key (# bits different in cyphertext)
0	1	0
1	6	2
2	21	14
3	35	28
4	39	32
5	34	30
6	32	32
7	31	35
8	29	34
9	42	40
10	44	38
11	32	31
12	30	33
13	30	28
14	26	26
15	29	34
16	34	35

Changes avalanche quickly

No similarity between original and new cyphertext

(half the bits in cyphertext flipped on average)