计算机网络安全

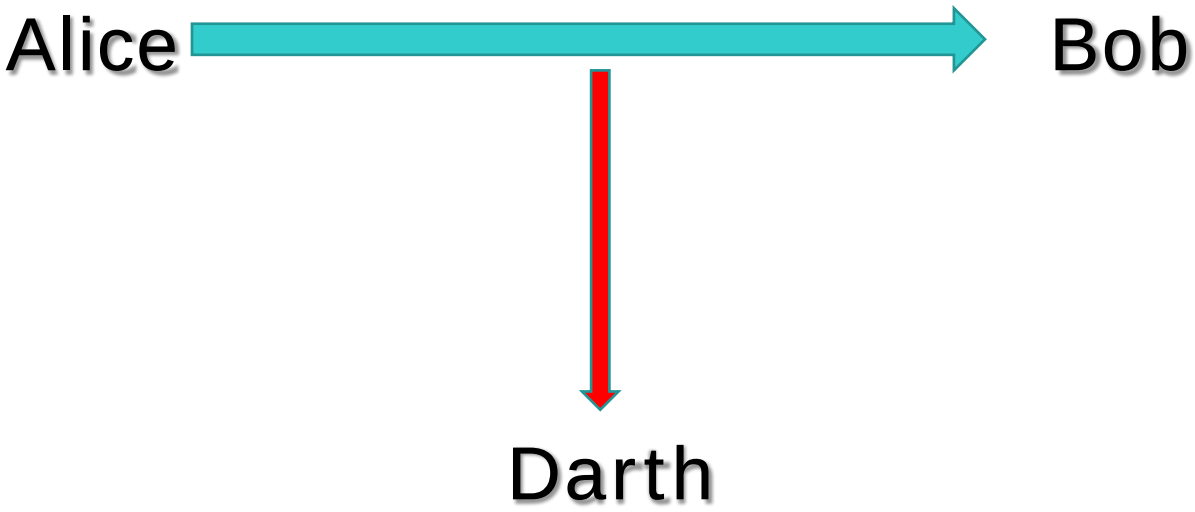
第一部分 密码学

第2章 对称加密和消息机密性

主要内容

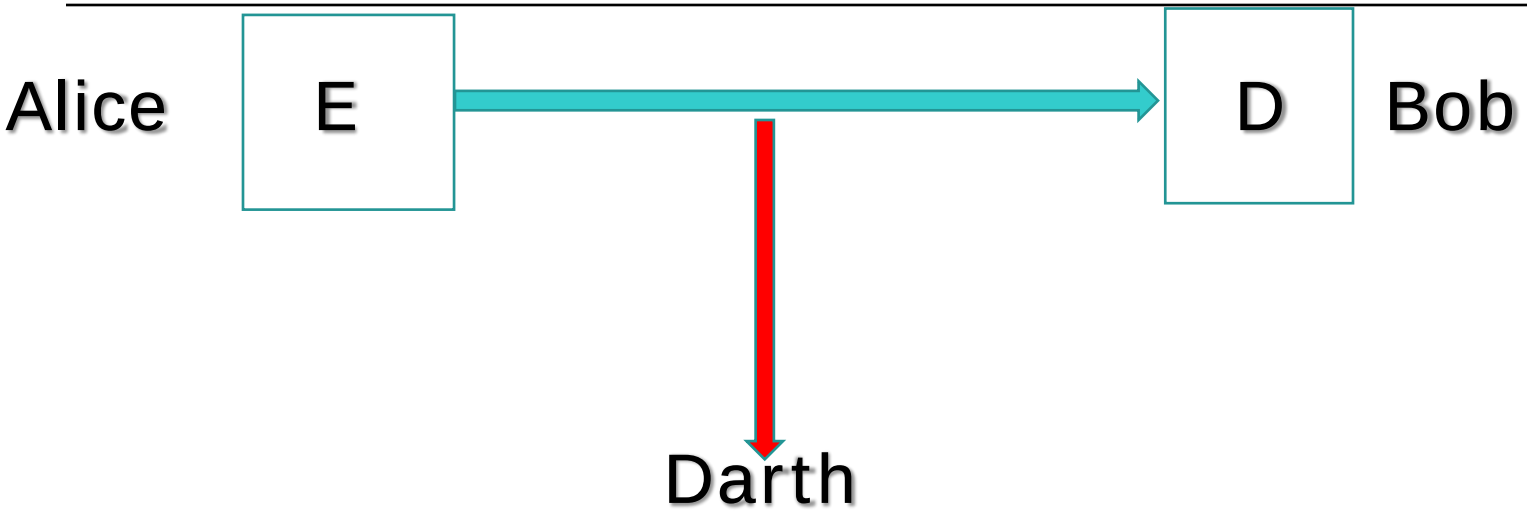
- 2.1 密码学简介
- 2.2 古典密码学
- 2.3 对称密码学
- 2.4 随机数和伪随机数
- 2.5 流密码和RC4
- 2.6 分组密码工作模式

密码学介绍



- Alice给Bob发送消息
- Darth将窃取消息
- 为保证消息不被窃取，Alice使用什么方法让Oscar不能得到自己发出的消息？

密码学介绍



- Alice将自己发出的消息变换为不可读信息
- Bob使用对应的方法将不可读信息转变为原始明文
- Darth只能获取不可读信息
- 密码学就是如此防止（破解）两者间通信被第三方截获的学科

网络信息安全基石—密码学

- 思考：公司员工薪资保密问题
- 思考：百万富翁斗富问题
- 思考：3次方程求根发明权问题
  - 塔尔塔利亚、菲奥尔
- 思考：Bob和Alice扫地问题?
- 思考：ABC * 91 = DE136，求ABC?
- 经典密码学算法：MD5、SHA、DES、AES、RSA、ECC

计算机网络安全

2.1 密码学简介

2.1.1 密码学发展历史

- 1.古典密码学
- 2.现代密码学

1. 古典密码学

- 起始时间：从古代到19世纪末，长达几千年，直到20世纪50年代。
- 密码体制：纸、笔或者简单器械实现的简单替代及换位
- 通信手段：信使
- 例子：行帮暗语、隐写术、藏头诗

2. 现代密码学

- 起始时间：从20世纪50年代至今
- 密码体制：分组密码、序列密码（即流密码）以及公开密钥密码，有坚实的数学理论基础。
- 通信手段：无线通信、有线通信、计算网络等

现代密码学的重要事件

- 1949年Shannon发表题为《保密通信的信息理论》，为密码系统建立了理论基础，从此密码学成了一门科学。（第一次飞跃）
- 1976年后，美国数据加密标准（DES）的公布使密码学的研究公开，密码学得到了迅速发展。
- 1976年，Diffe和Hellman提出公开密钥的加密体制的思想，1978年由Rivest、Shamire和Adleman 提出第一个比较完善的公钥密码体制算法（第二次飞跃）

（现代）密码学的基本概念

- 密码学（Cryptology）是结合数学、计算机科学、电子与通讯等诸多学科于一体的交叉学科，是研究密码编制和密码分析的规律和手段的技术科学。
- 密码学不仅具有信息通信加密功能，而且具有数字签名、身份认证、安全访问等功能
- 密码学提供的只是技术保障作用

2.1.2 密码学分支

- 密码学的两个分支
  - 1. 密码编码学
    - 研究密码变化的客观规律，设计各种加密方案，编制密码以保护信息安全
  - 2. 密码分析学
    - 研究如何分析和破译密码
  - 相互对立、相互促进

2.1.3 密码学的基本思想

- 用秘密方式写东西的艺术
- 对机密信息进行伪装
  - 将机密信息表述为不可读的方式
  - 有一种秘密的方法可以读取信息的内容

2.1.4 密码系统/密码体制

- **密码系统**
  - **一个用于加/解密，能够解决网络安全中的机密性、完整性、可用性、可控性和真实性等问题中的一个或几个的系统，也叫做密码体制**
- **由明文、密文、加密算法和解密算法、密钥五部分组成。**
  - **明文**：需要加密的原始信息，用m（即消息，Message)或P(即明文，Plain text)表示
  - **密文**：明文经过变换或伪装，形成密文， 用C表示（即密文，Cipher text)

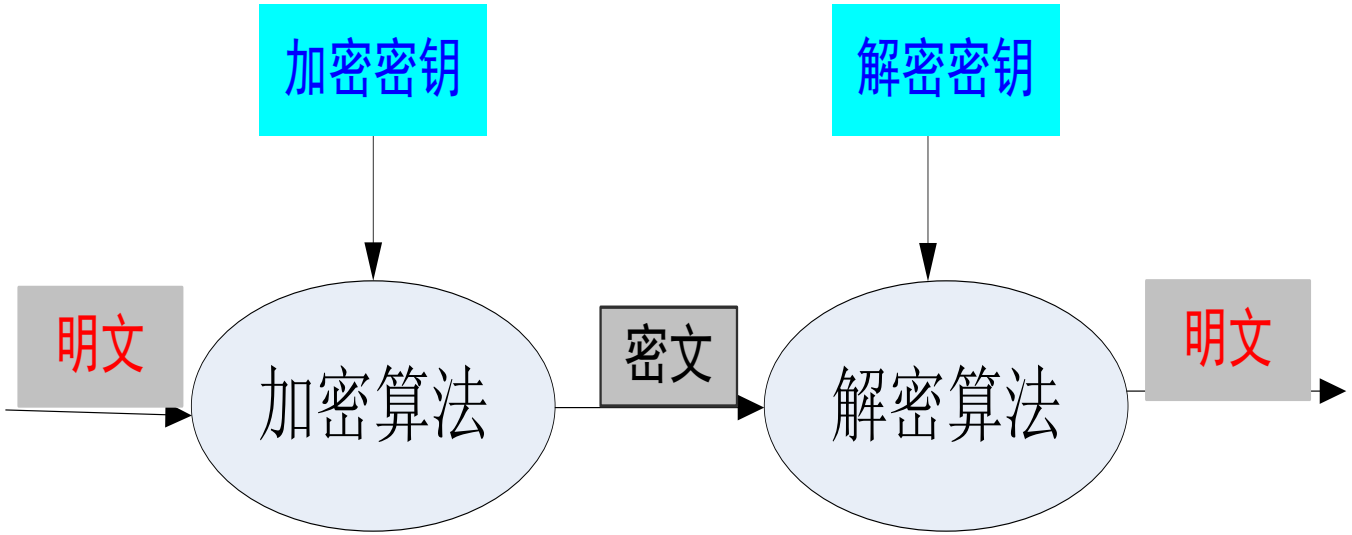
2.1.4 密码系统/密码体制（续）

- 密码算法
  - 加密变换与解密变换的具体规则
  - 商业上多公开，而军事上多保密
- 密钥空间
  - 加密和解密算法的操作是在密钥k控制下进行的。密钥的全体称为密钥空间
- 加密和解密
  - 加密(Encryption)：对明文实施的一系列变换过程 $E_k(m)$
  - 解密(Decrypt)：对密文施加的一系列的逆变换还原明文的过程 $D_k(c)$

1. 密码体制分类

- 对称密码体制（Symmetric Syetem）
  - 加密密钥和解密密钥相同，或者虽然不相同，但由其中的任意一个可以很容易地推出另一个，又称传统密码体制、秘密密钥体制或单密钥体制。
- 非对称密码体制（Asymmetric System）
  - 加密密钥和解密密钥不相同，并且从一个很难推出另一个，又称公开密钥体制
  - 用一个密钥进行加密，而用另一个进行解密。其中一个密钥可以公开，成为公开密钥(pulic key)，简称公钥；另一个密钥成为私人密钥（private key）,简称私钥。

2. 密码系统的模型



密码系统的安全性是基于密钥而不是加密和解密算法的细节。这意味着算法可以公开，甚至可以当成一个标准加以公布。

3. 密码系统的安全性

- **可破译**：由密文推出明文或密钥，或者由明文和密文可以寻求密钥
- **无条件安全性**：理论上不可破译
  - 不论提供的密文有多少，密文中所包含的信息都不足以惟一地确定其对应的明文；
  - 具有无限计算资源（诸如时间、空间、资金和设备等）的密码分析者也无法破译某个密码系统。

3. 密码系统的安全性（续）

- 计算安全性
  - 理论上可破译，但是需要付出十分巨大的计算，不能在希望的时间或可行的经济条件下求出准确的答案

4. 密码体制的基本原则

- 密码体制是不可破的（理论上不可破，计算上不可破）；
- 密码体制的安全性是依赖**密钥的保密**，而不是依赖于对加密体制的保密；
- 加密和解密算法适用于密钥空间中的所有元素；
- 密码体制既易于实现又便于使用。
- 密钥空间应足够大，使得试图通过穷举密钥空间进行搜索的方式在计算上不可行。

2.1.5 密码分析/密码攻击

- 借助窃听到的密文以及其它信息，通过各种方法推求原来的**明文**甚至**密钥**的过程，叫做**密码分析或密码攻击**
- **被动攻击**：窃听、分析，破坏明文信息的机密性
  - 防范措施：加密
- **主动攻击**：篡改、伪装等，破坏明文信息的完整性
  - 防范措施
    - 认证
    - 数字签名

1. 密码攻击方法

- 密码分析：常用的方法有以下4类：
  - 惟密文攻击
  - 已知明文攻击
  - 选择明文攻击
  - 选择密文攻击

1. 密码攻击方法（续）

- 唯密文攻击
  - 密码破译者除了拥有截获的密文，以及对密码体制和密文信息的一般了解外，没有什么其它可以利用的信息用于破译密码。
  - 在这种情况下进行密码破译是最困难的，经不起这种攻击的密码体制被认为是完全不保密的。
- 已知明文攻击
  - 密码破译者不仅掌握了相当数量的密文，还有一些已知的<明文，密文>对（通过各种手段得到的）可供利用。
    - 比如，电子金融消息往往有标准化的文件头或者标志
    - Imitation Game
  - 现代的密码体制（基本要求）不仅要经受得住唯密文攻击，而且要经受得住已知明文攻击。


THE TRUE ENIGMA
WAS THE MAN WHO CRACKED
THE CODE
BENEDICT CUMBERBATCH
KEIRA KNIGHTLEY
THE IMITATION GAME
COMING SOON
STUDIOCANAL

1. 密码攻击方法（续）

- 选择明文攻击
  - 密码破译者不仅能够获得一定数量的<明文，密文>对，还可以选择特定的明文，并通过某种手段（如欺骗）得到对应的密文。
- 选择密文攻击
  - 密码破译者能选择不同的被加密的密文，并还可得到对应的解密的明文，据此破译密钥及其它密文。

计算机网络安全

2.2 古典密码学

2.2.1 古典密码体制

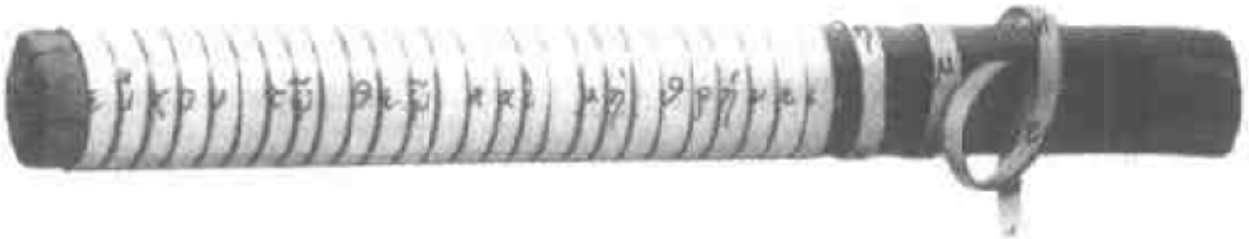
- 古典密码体制是指那些比较简单的、大多数采用**手工或机械操作**对明文进行加密、对密文进行解密的密码体制。
- 古典密码技术以**字符**为基本加密单元，大都比较简单
- 反映了密码设计和破译的思想，是学习密码学的基本入口，对于理解、设计和分析现代密码仍然具有借鉴的价值。

古典密码

- 凯撒密码

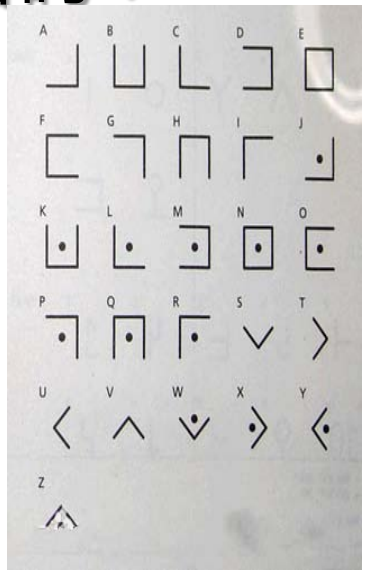
明文字母	abcdefghijklmnopqrstuvwxyz
密文字母	DEFGHIJKLMNOPQRSTUVWXYZABC

- 斯巴达人“天书”密码

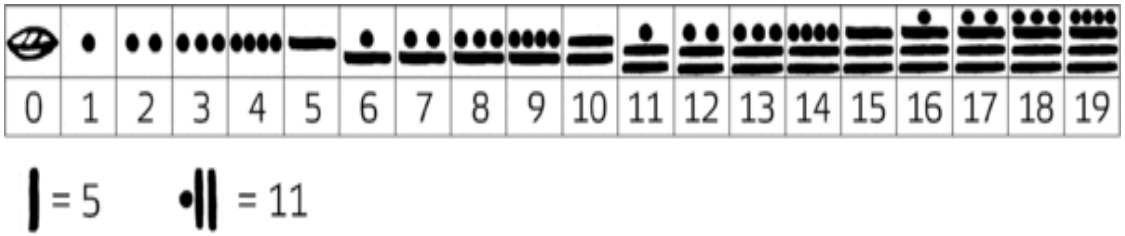


古典密码

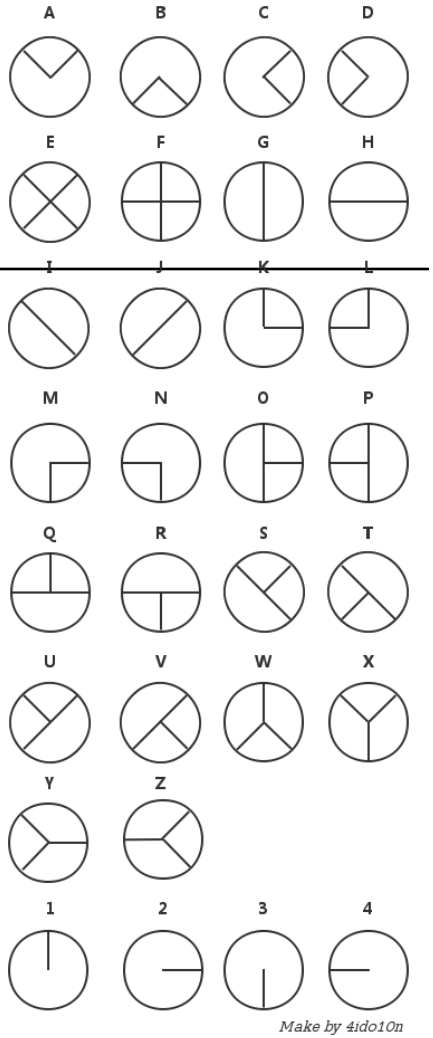
使用图案代替文字内容，例如猪圈密码：

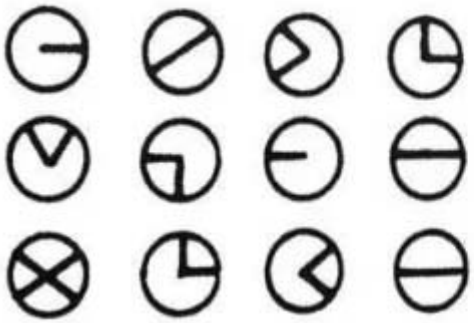


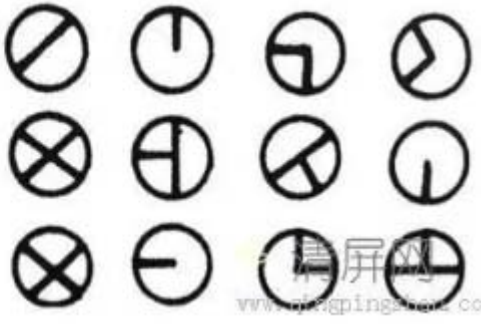
玛雅密码：



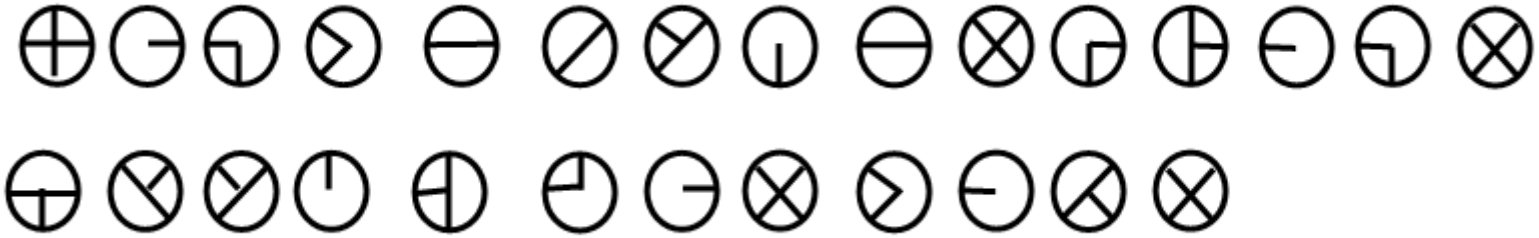
古典密码（夏多密码）







I AM IN DANGER SEND HELP



2.2.2 传统加密的基本方法

- 1. 置换
  - 对明文字符按某种规律进行位置的交换而形成新的排列
- 2. 代换
  - 将明文字母替换成其他字母、数字或符号的方法

1. 置换技术

- **一维倒置**
  - **报文倒置：将一整段明文整体倒置，并截成固定长度的字母组，形成密文**

    明文：**i am a student in nuist**

    密文：**tsiu nnit nedu tsam ai**

  - **分组倒置**：首先将明文分成固定长度的字符串，再对每组明文的字符分别进行倒置，形成密文

1. 置换技术（续）

- 二维易位
  - 首先根据约定的书写顺序填入明文，再按照约定的抄写顺序读出密码
  - 比如，按行写入，按列读出

明文：**i am a student in nuist**

i	a	m	a	s	t
u	d	e	n	t	i
n	n	u	i	s	t

密文：**iun adn meu ani sts tit**

2. 代换技术

- Kaesar密码(恺撒密码)
  - 对字母表中的每个字母用它之后的第3个字母来代换

    明文 meet me after class
    密文 PHHW PH DIWHU FODVV

- 如果让每个字母等价于一个数值（如：a,b,… ,z，等价数值为：0，1，…，25），则
  - 加密算法：C=E(3，P)=(P + 3) mod 26
  - 解密算法：P=D(k，C)=(C - k) mod 26

2. 代换技术（续）- 单表代换

加密函数

明文 →	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
代换为 →	D	K	V	Q	F	I	B	J	W	P	E	S	C	X	H	T	M	Y	A	U	O	L	R	G	Z	N

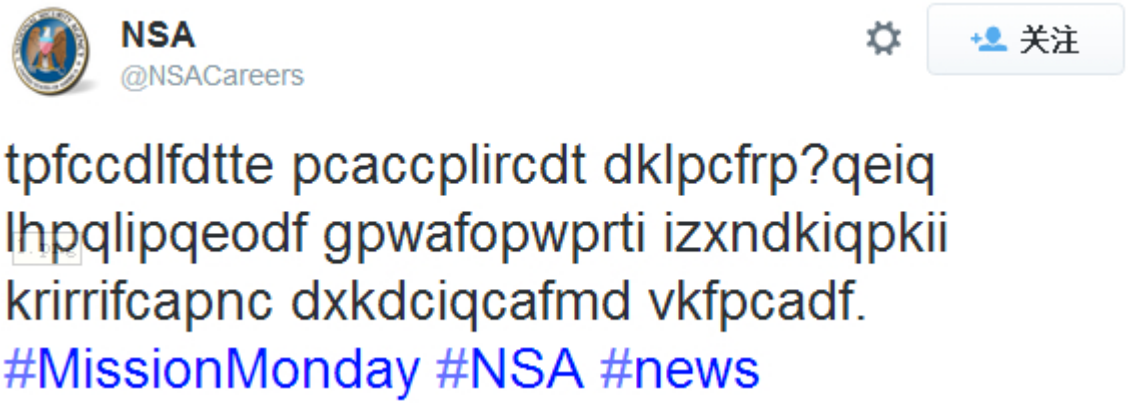
解密函数

密文 →	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
解密为 →	s	g	m	a	k	e	x	o	f	h	b	v	q	z	u	j	d	w	l	p	t	c	i	n	r	y

明文: if we wish to replace letters

密文: WI RF RWAJ UH YFTSDVF SFUUFYA

NSA Career密码



2. 代换技术（续）- 多表密码

- 多表密码是利用多个单表代换密码构成的密码体制。它在对明文进行加密的过程中依照密钥的指示轮流使用多个单表代替密码。
- $M=(m_1,m_2,\ldots,m_n)$ $K=(k_1,k_2,\ldots,k_n)$ $C=(c_1,c_2,\ldots,c_n)$
- 加密变换：$c_i=E_i(m_i)=m_i+k_i \bmod N$
- 解密变换： $m_i=D_i(m_i)=c_i - k_i \bmod N$

多表密码

K=(1,2,3), 明文长度超过密钥时，循环使用密钥。

K=1 →

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A

K=2 →

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B

K=3 →

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C

明文：**nuist**　　密文：**OWLTV**

明文中的字母“n”从k=1的对应表中查出“O”；而“u”则从k=2的表中查，依此类推。

多表代换密码举例

多表代换密码中最著名的一种密码称为**维吉尼亚（Vigenere）密码**。这是一种以移位代换为基础的周期代换密码，m个移位代换表由m个字母组成的密钥字确定（这里假设密钥字中m个字母不同，如果有相同的，则代换表的个数是密钥字中不同字母的个数）。

多表代换密码举例（续）

例：如果密钥字为deceptive，明文为we are discovered save yourself的加密过程为：

字母：a b c d e f g h i j k l m n o p q r s t u v w x y z

数码：0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25

明文：w e a r e d i s c o v e r e d s a v e y o u r s e l f

密钥：d e c e p t i v e d e c e p t i v e d e c e p t i v e

移位：3 4 2 4 15 19 8 21 4 3 4 2 4 15 19 8 21 4 3 4 2 4 15 19 8 21 4

密文：Z I C V T W Q N G R Z G V T W A V Z H C Q Y G L M G J

加密过程：密钥字母d对应数字3，因而明文字母w在密钥字母d 的作用下向右移3位，得到密文字母Z；依此类推。

2. 代换技术（续）-多字母代换密码

前面介绍的密码都是以单个字母作为代换的对象，对多于一个字母进行代换，就是多字母代换密码。它的优点是容易将字母出现的频度隐蔽，从而抗击统计分析。这里介绍Hill密码，它是数学家Lester Hill于1929年研制的。虽然这类密码由于加密操作复杂而未能广泛应用，但仍在很大程度上推进了经典密码学的研究。

Hill密码将明文分成每m个字母为一组的明文组，若最后一组不够m个字母就用字母补足，每组用m个密文字母代换，这种代换由m个线性方程决定，其中字母a、b、、y、z分别用数字0、1.、24. 25表示。

多字母代换密码（续）

若m=3，该系统描述如下：

（C表示密文，P表示明文，K表示密钥）

$$C_1=(k_{11}P_1+k_{12}P_2+k_{13}P_3) \bmod 26$$

$$C_2=(k_{21}P_1+k_{22}P_2+k_{23}P_3) \bmod 26$$

$$C_3=(k_{31}P_1+k_{32}P_2+k_{33}P_3) \bmod 26$$

用列向量和矩阵表示：

$$\begin{bmatrix}C_1\\C_2\\C_3\end{bmatrix}=\begin{bmatrix}k_{11} & k_{12} & k_{13}\\k_{21} & k_{2\,2} & k_{23}\\k_{31} & k_{31} & k_{31}\end{bmatrix}\begin{bmatrix}P_1\\P_2\\P_3\end{bmatrix}$$

或 $C=KP$

多字母代换密码举例

例：用密钥

$$K=\begin{bmatrix}11 & 3\\ 8 & 7\end{bmatrix}$$

来加密明文july。将明文分成两个组ju和ly，分别为[9，20]和[11，24]，计算如下：

$$\begin{bmatrix}11 & 3\\ 8 & 7\end{bmatrix}\begin{bmatrix}9\\ 20\end{bmatrix}=\begin{bmatrix}99+60\ mod\ 26\\ 72+140\ mod\ 26\end{bmatrix}=\begin{bmatrix}159\ mod\ 26\\ 212 mod\ 26\end{bmatrix}=\begin{bmatrix}3\\ 4\end{bmatrix}$$

$$\begin{bmatrix}11 & 3\\ 8 & 7\end{bmatrix}\begin{bmatrix}11\\ 24\end{bmatrix}=\begin{bmatrix}121+72\ mod\ 26\\ 88+168\ mod\ 26\end{bmatrix}=\begin{bmatrix}193\ mod\ 26\\ 256\ mod\ 26\end{bmatrix}=\begin{bmatrix}11\\ 22\end{bmatrix}$$

因此 July 的加密结果为 delw。

为了解密，必须先计算密钥矩阵K的逆矩阵。

多字母代换密码举例（续）

$$K^{-1}=\begin{bmatrix} 7 & 23 \\ 18 & 11 \end{bmatrix}$$

计算：

$$P= K^{-1}C$$

$$\begin{bmatrix} 7 & 23 \\ 18 & 11 \end{bmatrix}\begin{bmatrix} 3 \\ 4 \end{bmatrix}=\begin{bmatrix} 21+92\ mod\ 26 \\ 54+44\ mod\ 26 \end{bmatrix}=\begin{bmatrix} 113\ mod\ 26 \\ 98\ mod\ 26 \end{bmatrix}=\begin{bmatrix} 9 \\ 20 \end{bmatrix}$$

$$\begin{bmatrix} 7 & 23 \\ 18 & 11 \end{bmatrix}\begin{bmatrix} 11 \\ 24 \end{bmatrix}=\begin{bmatrix} 77+506\ mod\ 26 \\ 198+242\ mod\ 26 \end{bmatrix}=\begin{bmatrix} 583\ mod\ 26 \\ 440\ mod\ 26 \end{bmatrix}=\begin{bmatrix} 11 \\ 24 \end{bmatrix}$$

得到明文： **July**

计算机网络安全

2.3 对称密码学

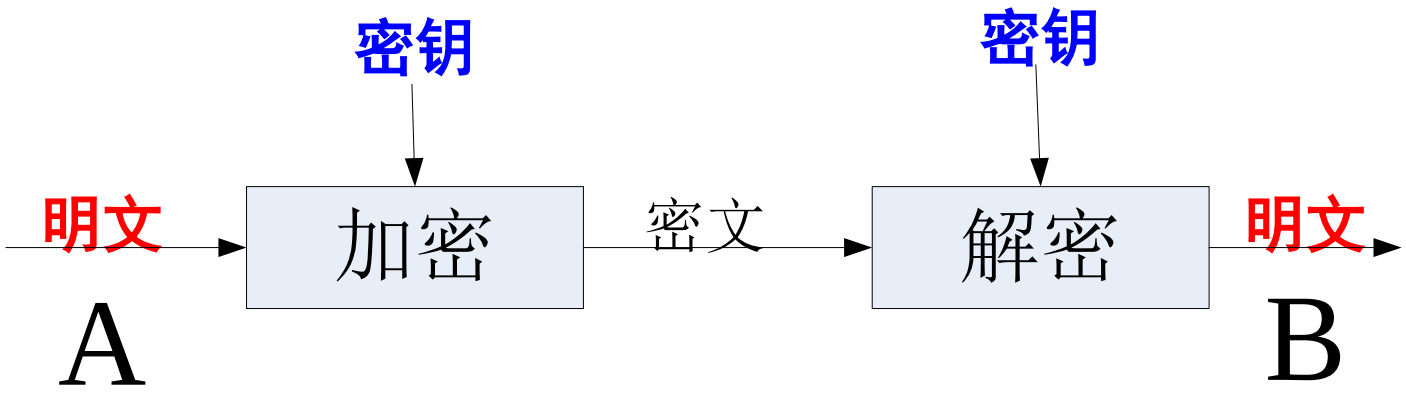
现代密码的算法

- 秘密密钥算法 - 对称密码体制
  - 加密和解密使用相同的密钥
- 公开密钥算法 - 公钥密码体制（非对称密码体制）
  - 不要求通信双方共享密钥，而是每个通信实体拥有两个密钥
    - 公钥：向其他人公开，用于加密
    - 私钥：不公开，用于解密
    - 不需要传递解密密钥 -私钥
- 哈希算法
  - 由任意长的消息计算出一个固定长度数值的数学变换：m → h(m)

2.3.1 秘密密钥算法（对称密码算法）

- 在算法中，加密和解密过程中的密钥相同。它要求发送者和接收者在安全通信之前，商定一个密钥。
- 又称为对称密码算法、传统密码算法
- 秘密密钥算法的安全性依赖于密钥，泄漏密钥就意味着任何人都能对消息进行加密解密。只要通信需要保密，密钥就必须保密。

1.对称密码算法的模型



○明文：原始信息。
○加密算法：以密钥为参数，对明文进行变换结果为密文。
○密钥：加密与解密算法的参数，直接影响对明文进行变换的结果。
○密文：对明文进行变换的结果。
○解密算法：加密算法的逆变换，以密文为输入、密钥为参数，变换结果为明文。

2. 对称密码算法的分类

- （1）流密码
- （2）分组密码

（1）流密码

- 流密码是对称密码体制中的一类，主要用于政府、军事等领域。
- 加密过程：先把明文转换成明文数据序列，然后同密钥序列进行**逐位加密**生成密文序列发送给接收者。
- 解密过程：用相同的密钥序列对密文序列进行逐位解密以恢复出明文序列。

（2）分组密码

- 分组密码是现代密码学中的重要体制之一，也是应用最为广泛、影响最大的一种密码体制。
- 分组密码的加密原理是将明文按照某一规定的n bit长度分组（最后一组长度不够时要用规定的值填充，使其成为完整的一组），然后使用相同的密钥对每一分组分别进行加密。

分组密码的设计思想

- 扩散
- 混乱
- 目标：使得对于不知道密钥的人来说，从明文到密文的映射看起来是随机的，以防止用统计的手段来破译

扩散

- 所谓扩散，是
  - 将算法设计得使每一比特明文的变化尽可能多地影响到输出密文序列的变化，以便隐蔽明文的统计特性；
  - 将每一位密钥的影响也尽可能迅速地扩展到较多的输出密文比特中去。
- 扩散的目的是希望密文中的任一比特都要尽可能与明文、密文相关联，或者说，明文和密钥中任何一比特的改变，对密文的每个比特都有影响，能够以50%的概率改变密文的每个比特

扩散的举例说明

- **无扩散技术的加密**

  p1:00000000　　c1:00000010

  p2:00000001　　c2:00000011

- **有扩散技术的加密**

  p1:00000000　　c1:01011010

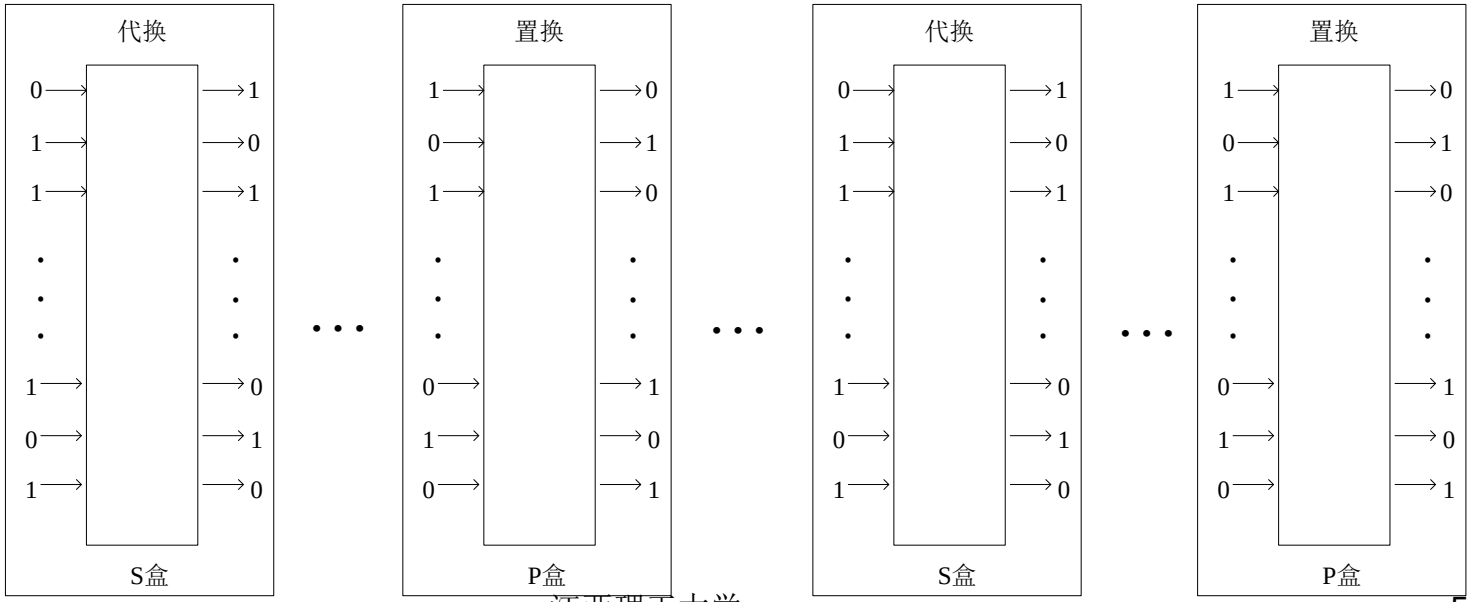
  p2:00000001　　c2:11101011

混乱

- 所谓混乱，是指在加密变换过程中使得明文、密钥以及密文之间的关系尽可能地**复杂化**，以防密码破译者采用统计分析法进行破译攻击。
- 混乱可以用“**搅拌机**”来形象地解释，将一组明文和一组密钥输入到算法中，经过充分混合，最后变成密文。
  - 但是，执行这种“混乱”作业的每一步都必须是可逆的，即明文混乱以后能得到密文，反之，密文经过逆向的混乱操作以后能恢复出明文。

分组密码常用技术：s-p网络

- Substitution-Permutation Network, 代换-置换网络
- s-p网络由s变换和p变换交替进行多次迭代，它属于迭代密码，也是乘积密码的常见表现形式。



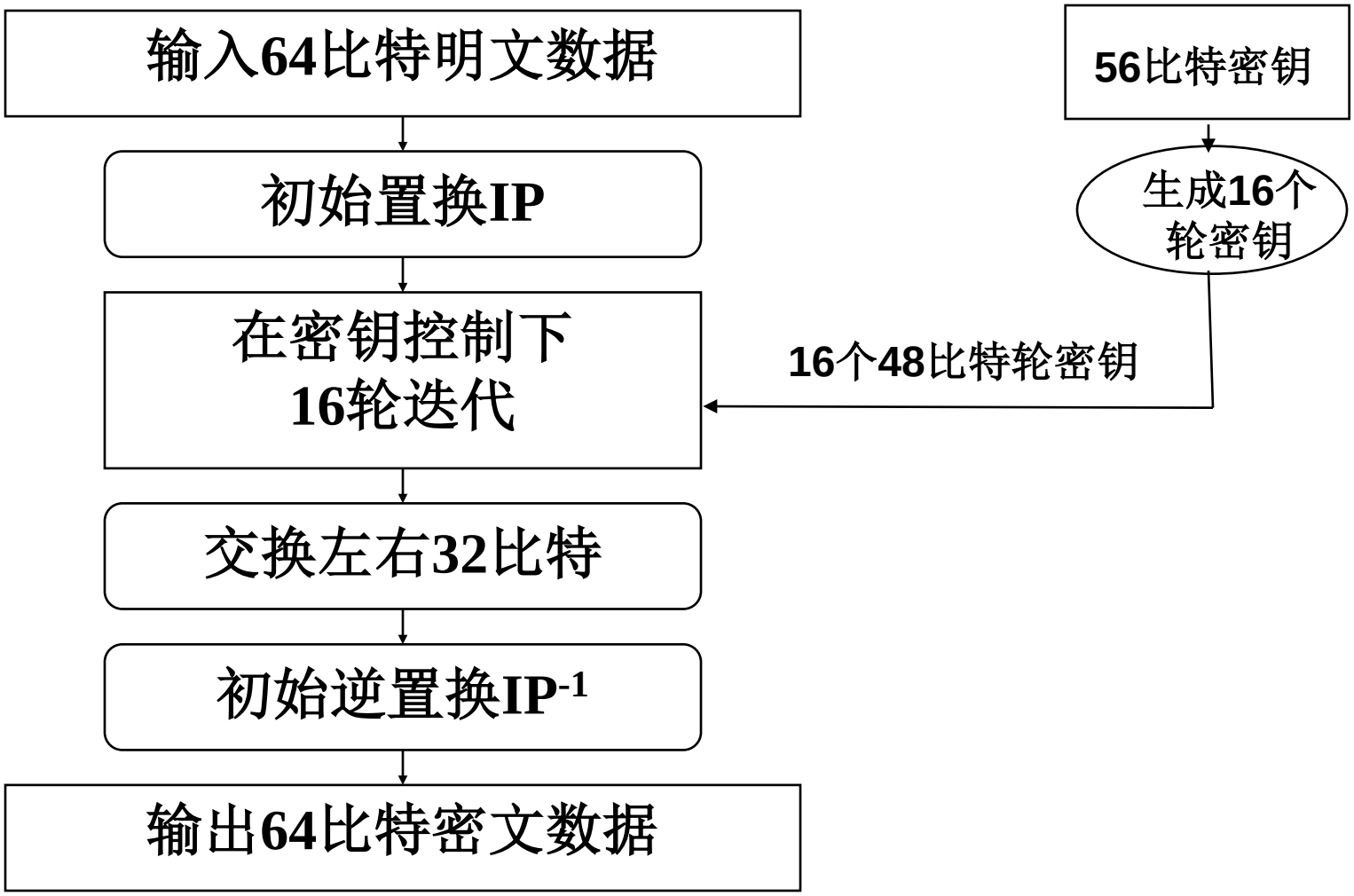
2.3.2 DES对称加密技术

- 数据加密标准（Data Encryption Standard, DES），1977年由美国国家标准局发布
- 分组加密算法：明文和密文为64位分组长度。
- 对称算法：加密和解密除密钥编排不同外，使用同一算法。
- 密钥长度：56位。看起来是64位，但每个第8位为奇偶校验位，可忽略。
- 采用混乱和扩散的组合，每个组合先代换后置换，共16轮。
- 只使用了标准的算术和逻辑运算，易于实现。
- DES得到了包括金融业在内的广泛应用。对DES安全性的研究也在不断继续。

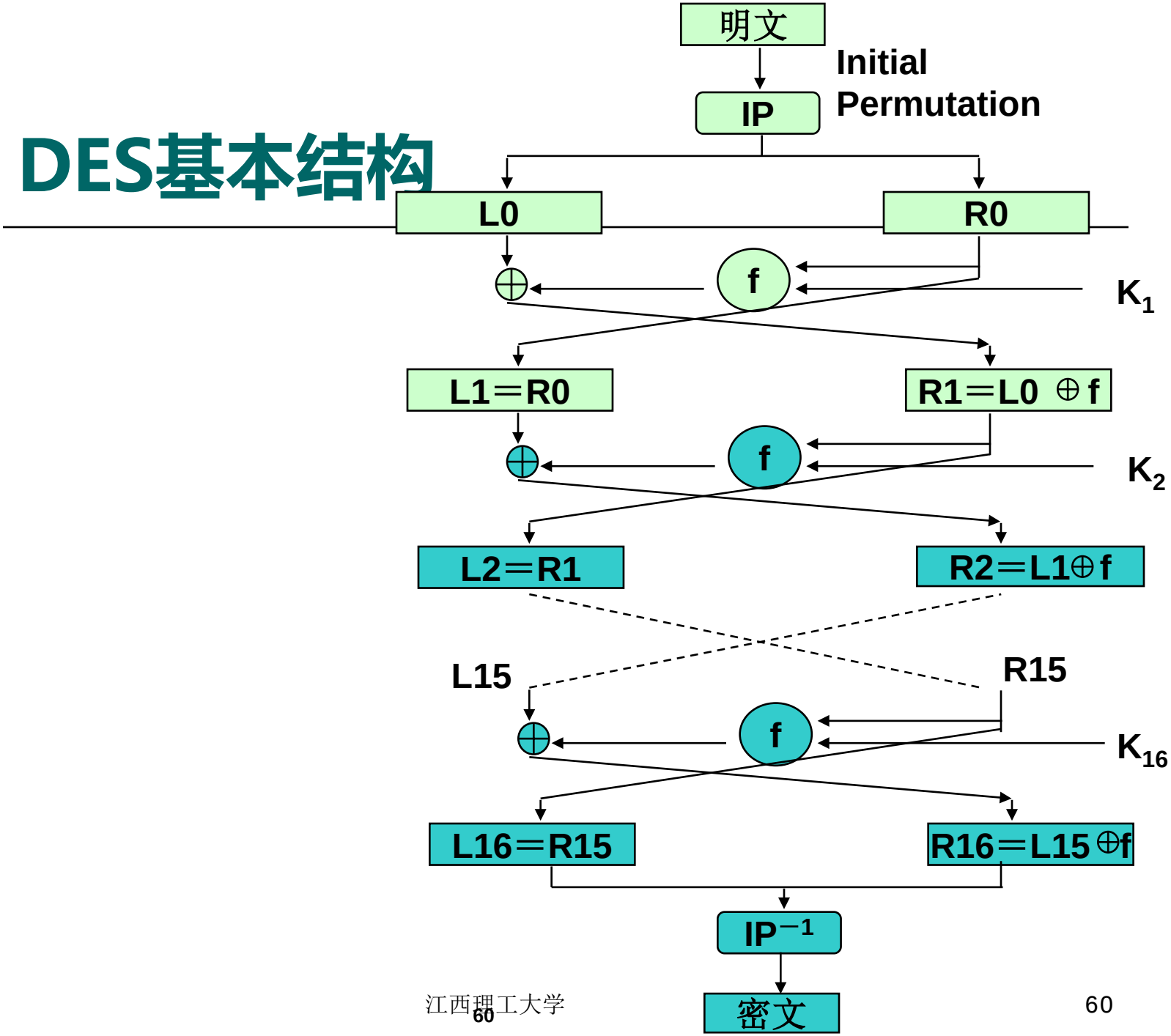
2.3.2 DES对称加密技术（续）

- 1998年5月美国研究机构EFF宣布用一台价值20万美元的计算机改装的专用解密系统，花费56小时破译了56位密钥的DES。
- 2000年10月2日，NIST公布了新的AES，DES作为标准正式结束。
- 尽管如此，学习DES，对于掌握分组密码的基本理论和设计思想仍然有重要参考价值。同时在非机密级的许多应用中，DES仍在广泛使用。

DES基本结构


输入64比特明文数据
初始置换IP
在密钥控制下
16轮迭代
交换左右32比特
初始逆置换IP-1
输出64比特密文数据
56比特密钥
生成16个
轮密钥
16个48比特轮密钥

DES基本结构


明文
Initial
Permutation
IP
L0
R0
f
K1
L1＝R0
R1＝L0 ⊕f
f
K2
L2＝R1
R2＝L1⊕f
L15
R15
f
K16
L16＝R15
R16＝L15⊕f
IP−1
密文

初始置换IP和初始逆置换IP^{-1}

IP置换：设输入$x=x_1x_2\ldots x_{64}$，则$IP(x)=x_{58}x_{50}\ldots x_7$输出

初始置换 IP

58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

位置1

位置40

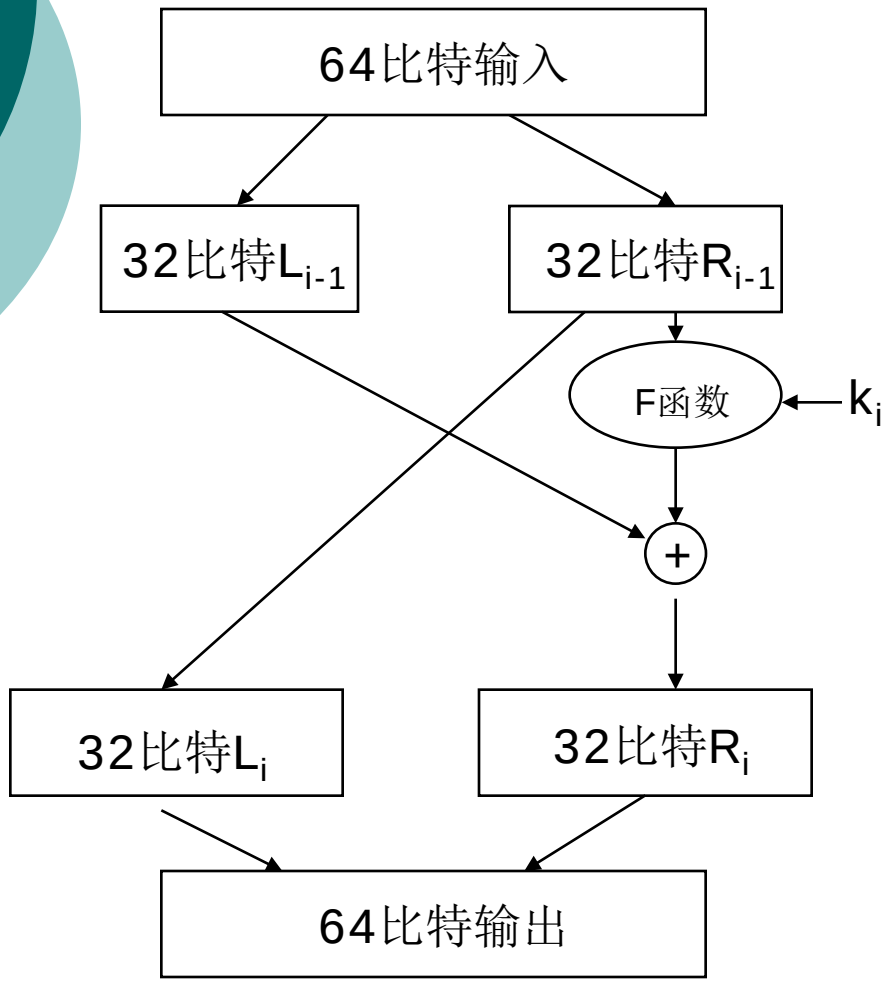
初始逆置换 IP^{-1}

40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

位置1

经过16次迭代运算后，作为输入，进行逆变换，即得到密文输出。例如，第1位经置换后，处于第40位，通过逆置换IP^{-1}，又将第40位挽回到第1位。

DES的一轮加密运算

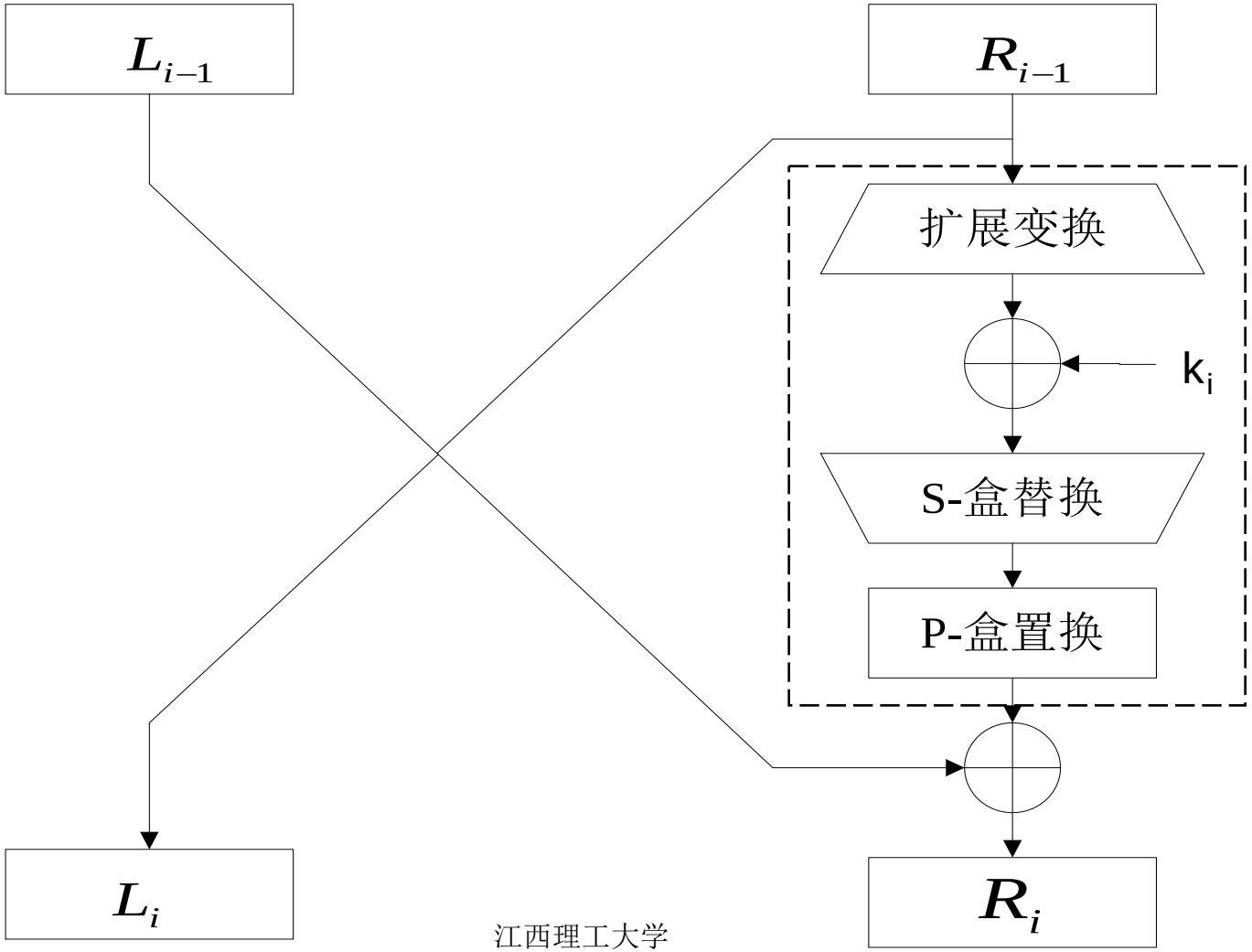


$$L_i = R_{i-1}$$

$$R_i = L_{i-1} \oplus f(R_{i-1}, K_i)$$

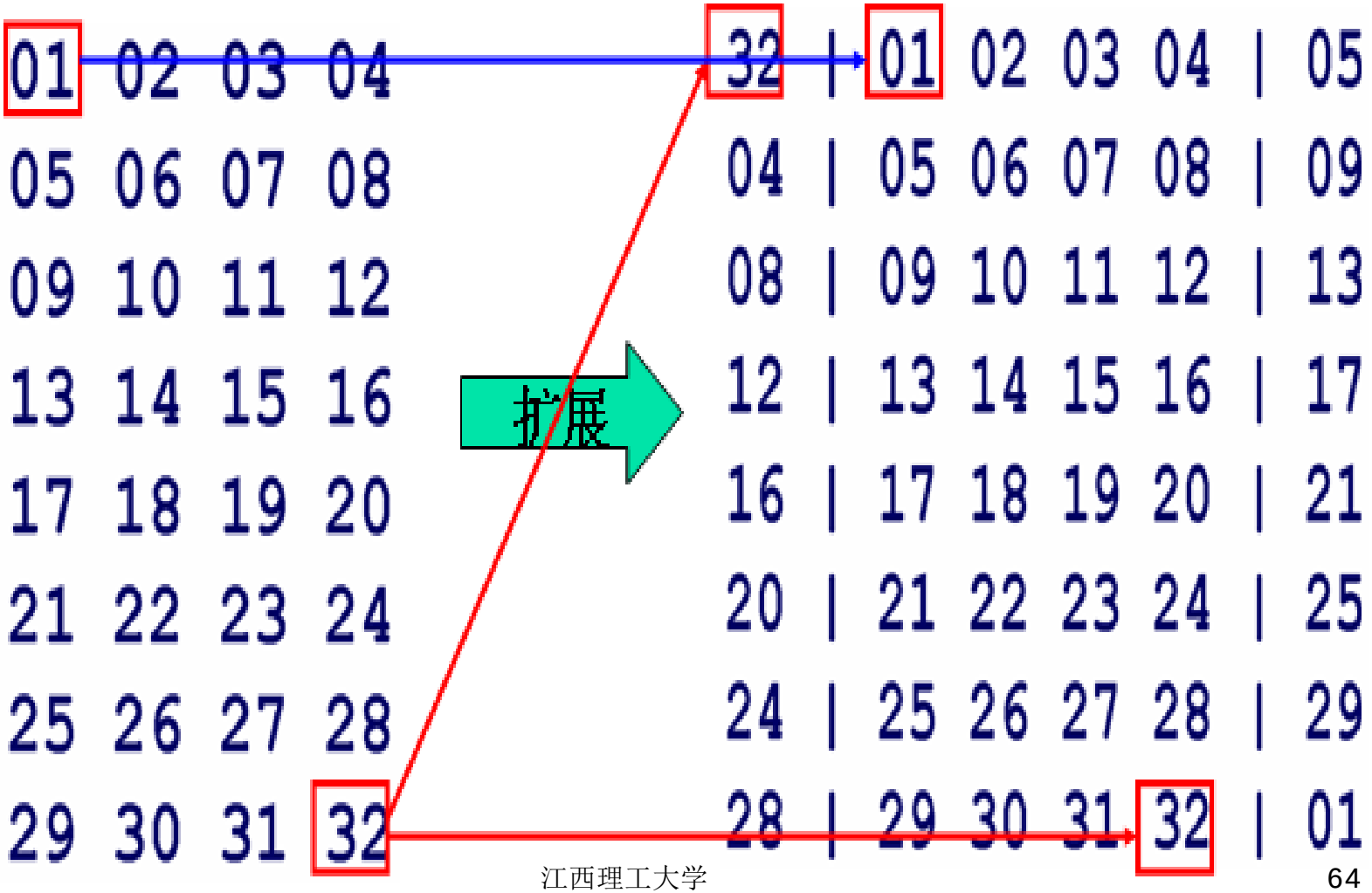
加密

F函数

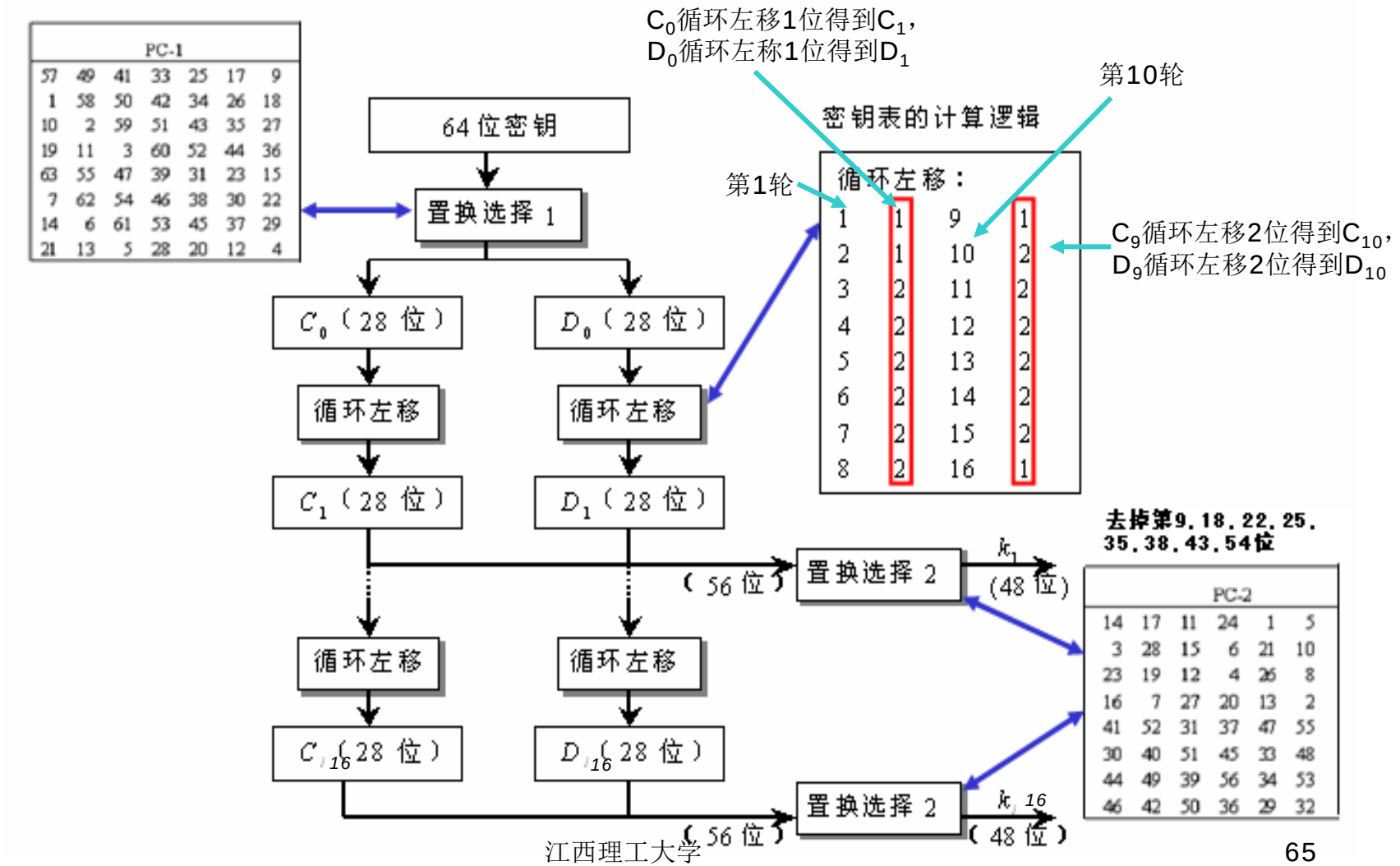


E扩展变换(将32比特扩展成48比特）

32比特数据 → 48比特数据



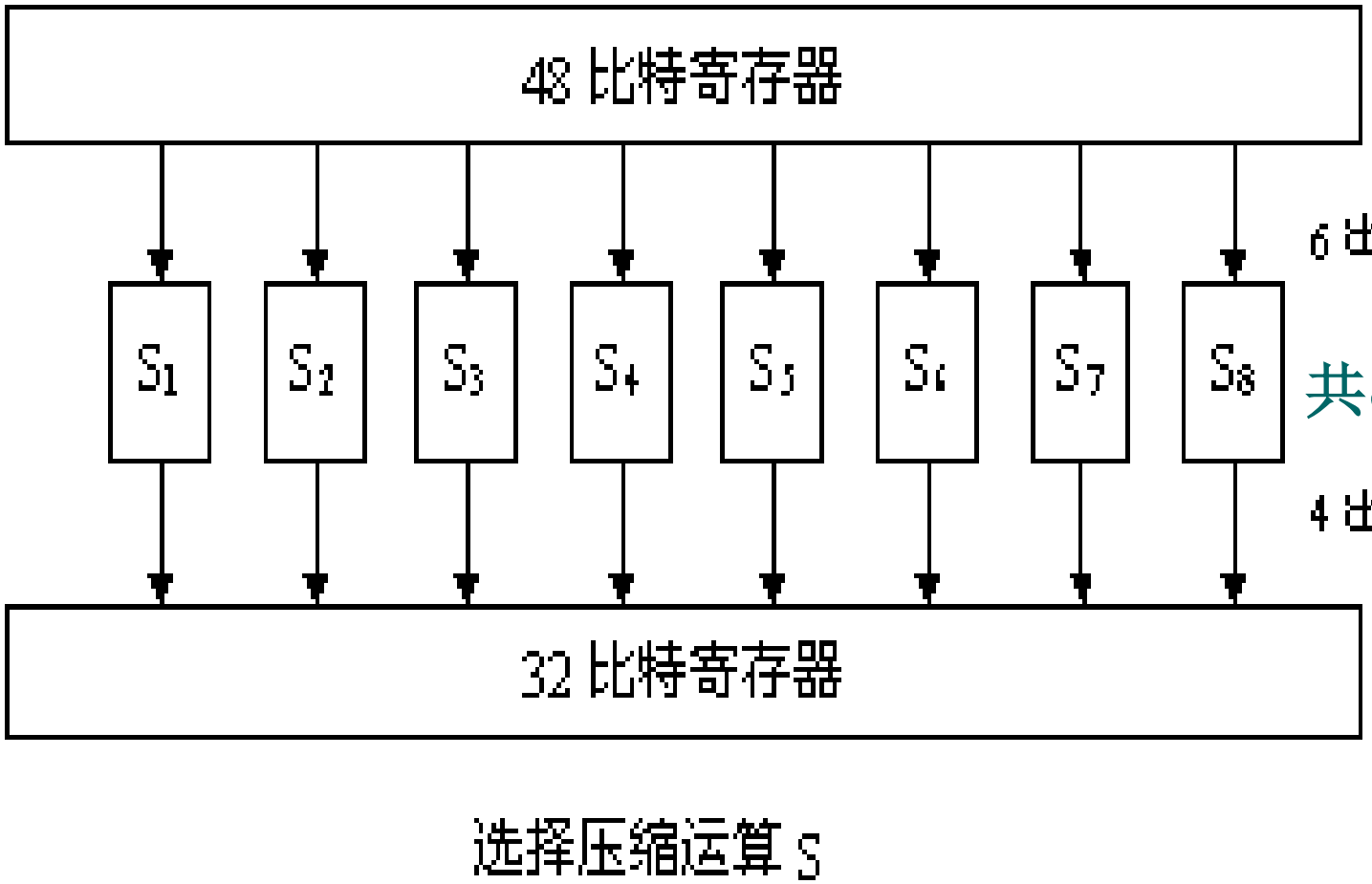
DES子密钥的生成


PC-1
57 49 41 33 25 17 9
1 58 50 42 34 26 18
10 2 59 51 43 35 27
19 11 3 60 52 44 36
63 55 47 39 31 23 15
7 62 54 46 38 30 22
14 6 61 53 45 37 29
21 13 5 28 20 12 4
64 位密钥
置换选择 1
C_0（28 位）
D_0（28 位）
循环左移
循环左移
C_1（28 位）
D_1（28 位）
循环左移
循环左移
C_{16}（28 位）
D_{16}（28 位）
（56 位）
置换选择 2
k_1
（48 位）
（56 位）
置换选择 2
k_{16}
（48 位）
C_0循环左移1位得到C_1，
D_0循环左称1位得到D_1
第10轮
密钥表的计算逻辑
第1轮
循环左移：
1 1 9 1
2 1 10 2
3 2 11 2
4 2 12 2
5 2 13 2
6 2 14 2
7 2 15 2
8 2 16 1
C_9循环左移2位得到C_{10}，
D_9循环左移2位得到D_{10}
去掉第9，18，22，25，
35，38，43，54位
PC-2
14 17 11 24 1 5
3 28 15 6 21 10
23 19 12 4 26 8
16 7 27 20 13 2
41 52 31 37 47 55
30 40 51 45 33 48
44 49 39 56 34 53
46 42 50 36 29 32

XOR操作

- R_{i-1}（扩展48位）XOR K_i（48位）
- 下一步，S盒替换

S盒替换



S盒的规则

14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

S-盒1

15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

S-盒2

10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

S-盒3

7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

S-盒4

2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

S-盒5

12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13

S-盒6

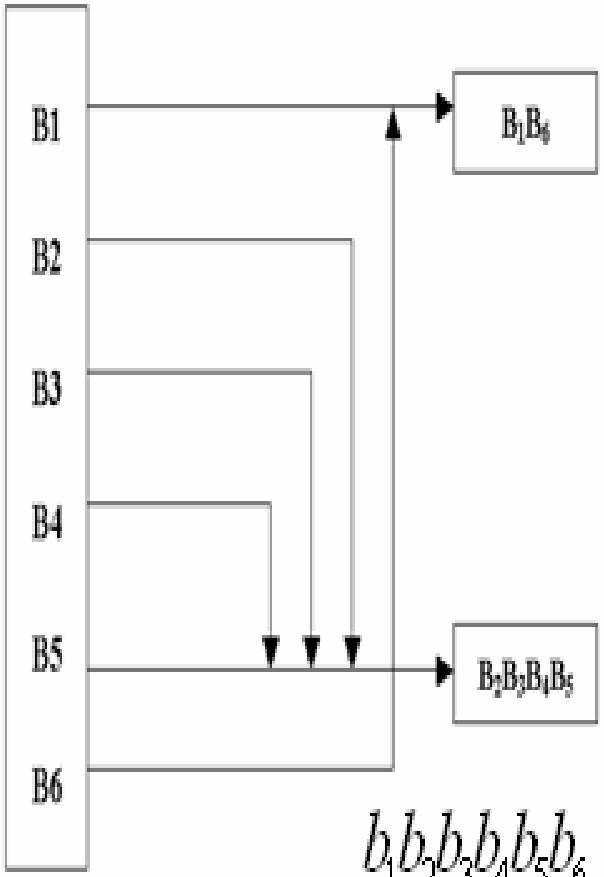
4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

S-盒7

13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
1	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

S-盒8

S-盒的构造



行

行\列	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
2	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
3	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13

列

$$\frac{b_1b_2b_3b_4b_5b_6}{110011} \Rightarrow \frac{\text{行：}\ b_1b_6 = 11_2 = 3}{\text{列：}\ b_2b_3b_4b_5 = 1001_2 = 9} \Rightarrow \frac{S_6\text{-盒子 3行9列}}{\text{值：}\ 14=1100}$$

P盒置换

16	07	20	21	29	12	28	17
01	15	23	26	05	18	31	10
02	08	24	14	32	27	03	09
19	13	30	06	22	11	04	25

保证上一轮某个s盒的输出对下一轮多个s盒产生影响.

将32比特数据重新排列（置换）。

DES解密

- 解密算法和加密算法相同
- 区别在于子密钥（轮密钥）使用次序相反
  - 加密：$K_1, k_2, k_3, \ldots k_{16}$
  - 解密：$k_{16}, \ldots k_3, k_2, k_1$

实验：用python进行DES加解密

- DES加密算法
  - key= ‘abcdefg’，
  - 明文 ‘Hello, welcome to jxust! ‘
  - 求密文，并解密

DES问题讨论

- DES的强度：56比特的密钥长度
  - 理论上的强度，97年$100000的机器可以在6小时内用穷举法攻破DES。
  - 实际攻破的例子，97年1月提出挑战，有人利用Internet的分布式计算能力，组织志愿军连接了70000多个系统在96天后攻破。
  - 这意味着随着计算能力的增长，必须相应地增加算法密钥的长度。
- 最近的一次评估是在1994年1月，当时决定1998年12月以后，DES不再作为联邦加密标准。
- 3DES和AES（128位）取代DES

2.3.3 三重DES加密技术

- 克服密钥长度较短而造成的不安全
- 使用3倍DES密钥长度的密钥，执行3次DES算法
- 四种模式，包括：
  - DES-EEE3模式，使用三个不同的密钥（k1, k2, k3），进行三次加密
  - DES-EDE3模式，使用三个不同的密钥（k1, k2, k3），采用加密-解密-加密模式
  - DES-EEE2模式，使用两个不同的密钥（k1= k3, k2），进行三次加密。
  - DES-EDE2模式，使用两个不同的密钥（k1= k3, k2），采用加密-解密-加密模式。

三重DES的优缺点

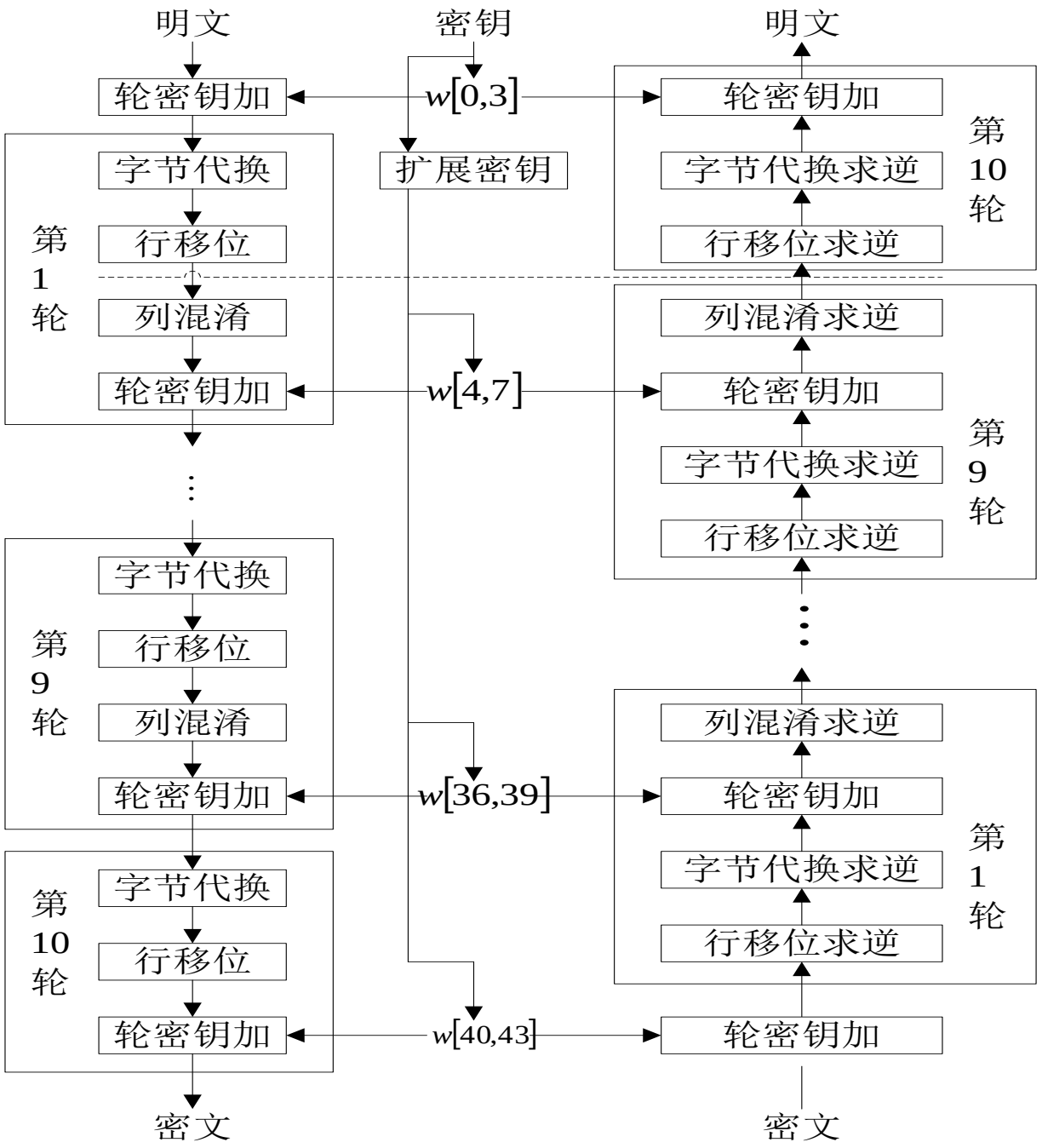
- 首先，密钥长度是112位（两个不同的密钥）或168位（三个不同的密钥），对抗穷举攻击的能力得到极大加强。
- 其次，3DES的底层加密算法与DES的加密算法相同。而DES算法是安全的。
- 缺点在于用软件实现该算法的速度比较慢
  - 这是因为DES一开始就是为硬件实现所设计的，难以用软件有效地实现。而3DES的底层加密算法与DES的加密算法相同，并且计算过程中轮的数量三倍于DES中轮的数量，故其速度慢得多。

2.3.4 AES加密技术

- 1997年，美国国家标准技术研究所(NIST)在全球范围内征集高级加密标准算法。
- 2002年10月， NIST宣布“Rijndael数据加密算法”最终入选，并将于2002年5月正式生效。
  - 实际上，目前通称的AES就是指的Rijndael对称分组密码算法。AES用来在将来取代DES，成为广泛使用的新标准。

AES结构

- AES算法也是迭代分组密码，明文分组长度有三个可选值，包括128、196、256比特，128位是使用最广泛的。密钥长度也有128、196、256比特三种，实现中也多取为128位。
- AES加解密的流程首先进行轮密钥加，然后进行完全相同的10轮迭代，得到最终输出


明文
密钥
明文
轮密钥加
w[0,3]
轮密钥加
第10轮
字节代换
扩展密钥
字节代换求逆
行移位
行移位求逆
第1轮
列混淆
列混淆求逆
轮密钥加
w[4,7]
轮密钥加
字节代换求逆
第9轮
行移位求逆
字节代换
行移位
第9轮
列混淆
列混淆求逆
轮密钥加
w[36,39]
轮密钥加
字节代换求逆
第10轮
字节代换
行移位求逆
行移位
第1轮
轮密钥加
w[40,43]
轮密钥加
密文
密文

(a)加密

(b)解密

AES256位加密解密练习

- KEY=venusCTF-hex IV=123-MD5
  现在请解密：
  a80d5eb43508e549f83e2e254c0a0f0644be58f453baced4af4777c4cd1b7575
  答案格式：ctfvenus{xxx}，所以答案是？

```
root@kali:~# python decodaes.py
ctfvenus{md5_aes_hex_all_i_like}
```

2.4 随机数和伪随机数

许多基于密码学的网络安全算法都使用随机数。

例子：

- 生成RSA公钥加密算法和其他公钥算法的密钥。
- 生成用作临时会话密钥的对称密钥; 用于许多网络应用程序，如传输层安全性，Wi-Fi，电子邮件安全性和IP安全性。
- 在许多密钥分发方案中，例如Kerberos，随机数用于握手以防止重放攻击。

对于一系列随机数的两个不同且不必要兼容的要求是：

- 随机性
- 不可预测性

2.4.1 随机数的应用

随机性

以下标准用于验证数字序列是否是随机的：

分布均匀

- 序列中的比特分布应该是均匀的。
- **1**和**0**的出现频率应大致相同。

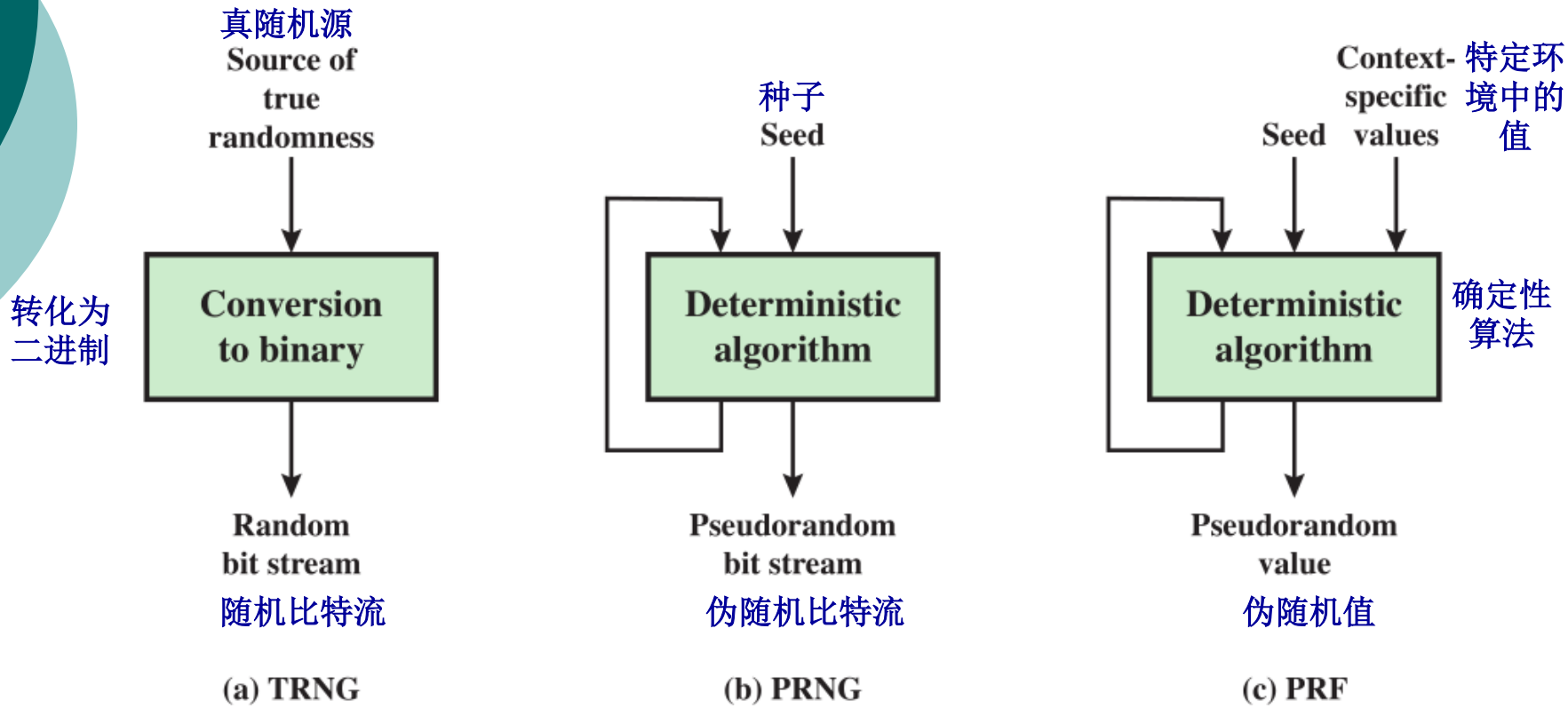
独立

- 序列中没有一个子序列可以从其他序列中推断出来。
- 没有一个标准来“证明”独立性。一般策略是应用一些测试，直到能确保满足独立性。

不可预测性

- **在诸如相互认证和会话密钥生成之类的应用中，并不要求数字序列在统计上是随机的，而是序列的连续成员是不可预测的。**
- **对于“真实的”随机序列，每个数字在统计上独立于序列中的其他数字，因此是不可预测的。**
- **必须注意的是，攻击者无法根据早期得到的数据预测序列的未来元素。**

2.3.2 真随机数发生器、伪随机数发生器和伪随机函数



TRNG = true random number generator （真随机数发生器）
PRNG = pseudorandom number generator （伪随机数发生器）
PRF = pseudorandom function （伪随机函数）

图2.6 随机和伪随机数发生器

2.3.3 算法设计

专用算法	• 专门设计用于生成伪随机比特流的目的
基于现有加密算法的算法	• 加密算法具有随机输入的效果 • 可以作为PRNG的核心
通常使用三大类加密算法来创建PRNG：	• 对称分组密码 • 不对称的密码 • 散列函数和消息认证码

2.5 流密码和RC4算法

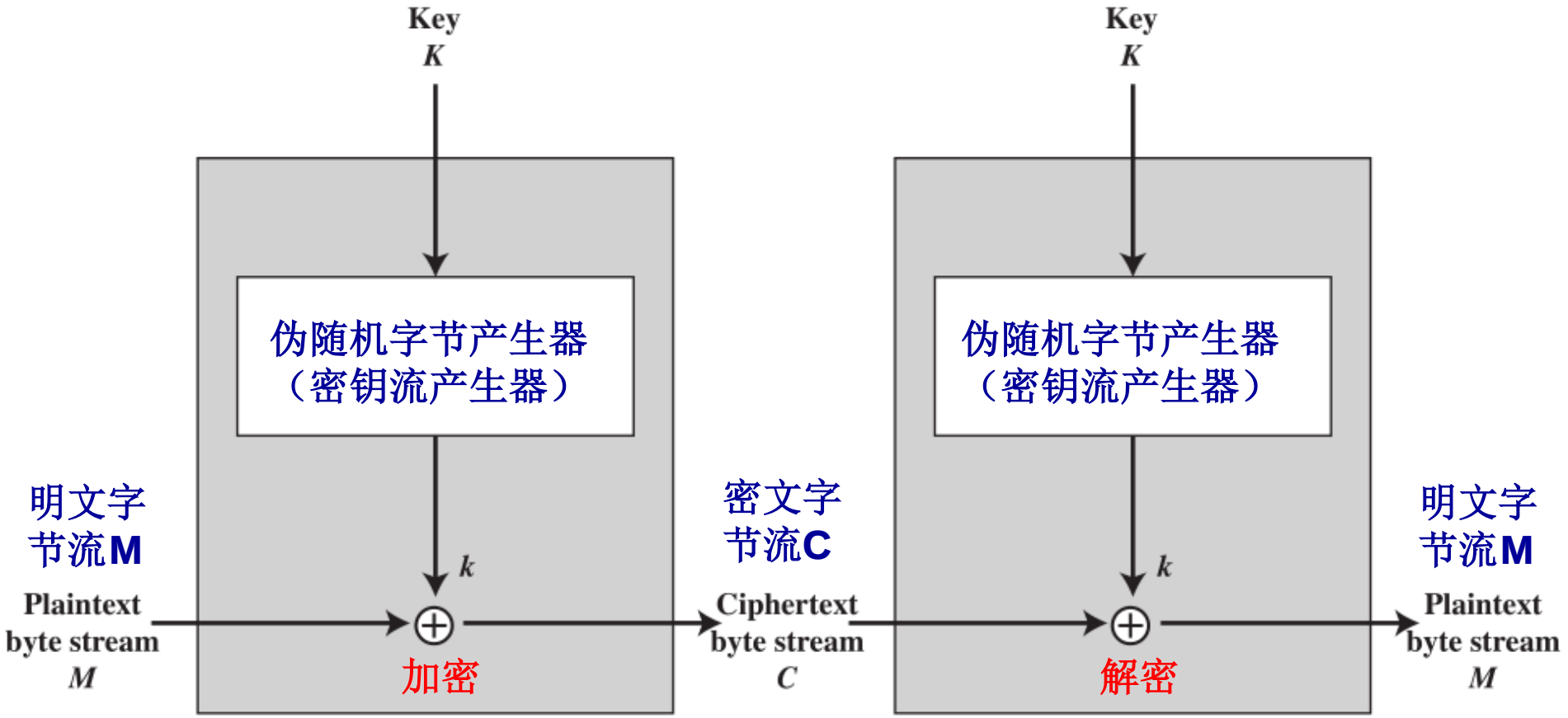
- **流密码(流密码是一次加密一比特位或一个字节的数据流的密码)**
  - 加密和解密每次只处理数据流的一个符号(如一个字符或一个比特)
  - 密钥输入到一个伪随机数(比特)发生器，该伪随机数发生器产生一串随机的8比特数，称为密钥流，通过与同一时刻一个字节的明文流进行异或(XOR)操作产生密文流。
  - 解密需要使用相同的伪随机序列，与密文相异或，得到明文。

2.5.1 流密码结构

- 典型的流密码一次加密一个字节的明文，如图2.7所示。
- 例如：如果伪随机数发生器产生的字节（k）是01101100，明文是11001100，那么得到的密文是：

加密		解密	
11001100	明文	10100000	密文
xor 01101100	密钥流	xor 01101100	密钥流
10100000	密文	11001100	明文

图2.7 流密码结构示意图



流密码设计需要考虑因素

- 加密序列应该有一个长周期。
  - 重复周期越长，进行密码分析就越困难。
- 密钥流应尽可能接近真随机数流的性质。
  - 密钥流越随机出现，密文越随机化，使密码分析更加困难。
- 伪随机数发生器的输出受输入密钥值控制。
  - 为防止蛮力攻击，密钥需要足够长。
  - 就当前科技水平而言，需要至少128位的密钥长度。

2.5.2 RC4算法

- **一种可变密钥长度的、面向字节操作的流密码。**
- **以一个足够大的表S为基础，对表进行非线性变换，产生密钥流。**
  - **一般S表取作256字节大小，用可变长度的种子密钥K（1到256个字节）初始化表S，S的元素记为S[0]，S[1]，S[255]。**
- **加密和解密的时候，密钥流中的一个字节由S中256个元素按一定方式选出一个元素而生成，同时S中的元素被重新置换一次。**

2.5.2 RC4算法（续）

- 初始化S。

开始时，S的元素按升序被置为0～255。T是创建的一个临时向量，keylen为字节长度的密钥。K为密钥。将K赋值给T的前keylen个元素，并循环使用K的值赋给T剩下的元素。

For i=0 to 255 do

S[i]=i;

T[i]=K[I mod keylen];

然后用T产生S的初始置换。

2.5.2 RC4算法（续）

- S的初始置换

```
j=0;
For i=0 to 255 do
   j=(j+S[i]+T[i]) mod 256;
 Swap(S[i], S[j]);
```

S就是置换，S仍然包含0～255的所有数。

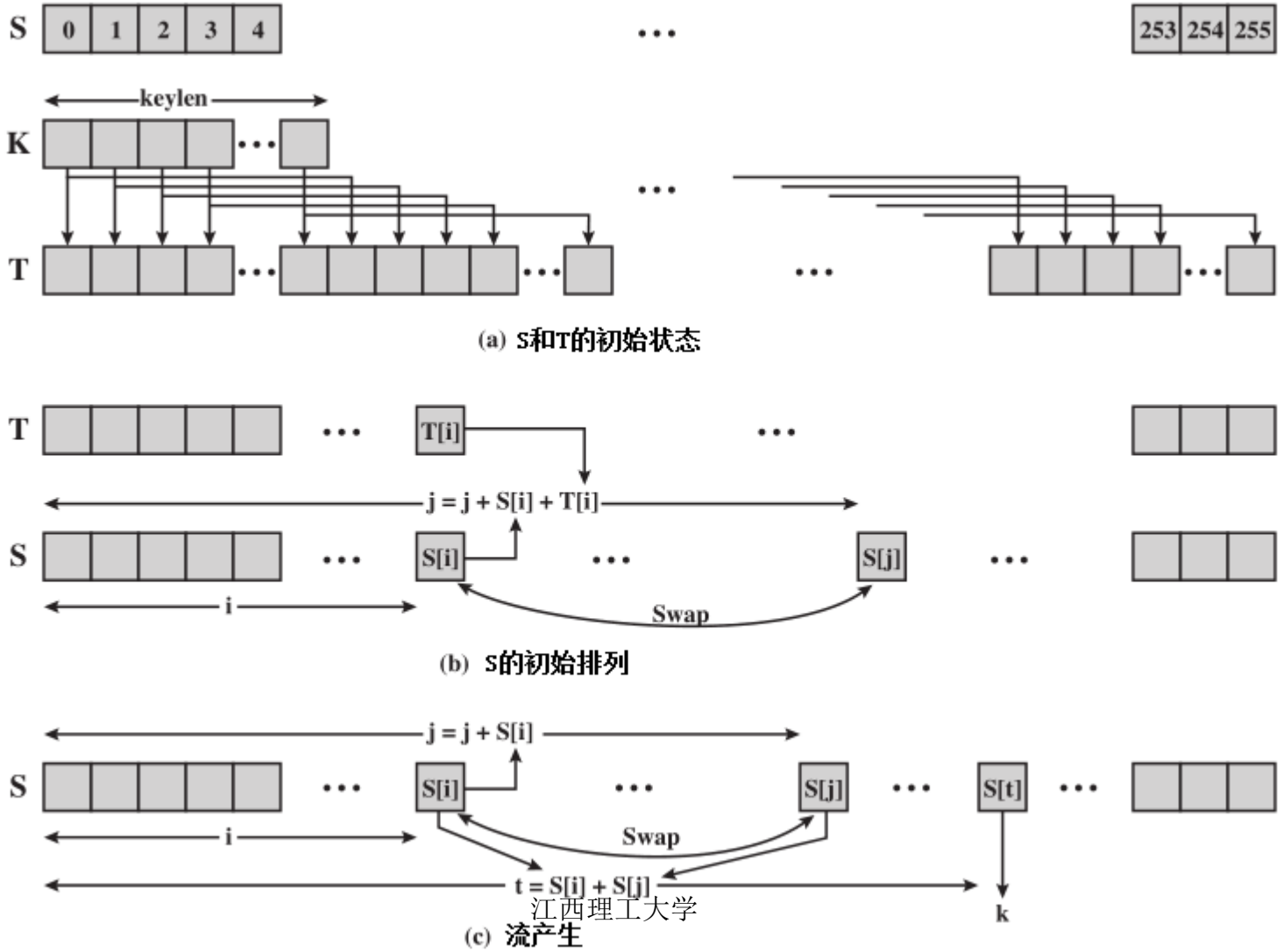
2.5.2 RC4算法（续）

- 流产生

```
i, j=0;
while(true)
    i=(i+1) mod 256;
    j=(j+S[i]) mod 256;
    Swap(S[i], S[j]);
    t=(S[i]+S[j]) mod 256;
    k=S[t];
```

k是一个8位二进制数密钥。将k值与明文的下一字节做异或。解密时，将k值与密文的下一字节做异或。

图2.8 RC4算法


S
0
1
2
3
4
253
254
255
keylen
K
T
(a) S和T的初始状态
T
T[i]
j = j + S[i] + T[i]
S
S[i]
S[j]
i
Swap
(b) S的初始排列
j = j + S[i]
S
S[i]
S[j]
S[t]
i
Swap
t = S[i] + S[j]
k
(c) 流产生

RC4加解密实例

- Key: “jxust”
- 明文： “Hello, this is a RC4 program”
- 密文：
  dbcc9da918b6e2b9c819baabd21b4ac0e8805ee7c7cb8013488e4757
- 或
- 厶漆□垛谷□韩?J黎€^缜薯□H圌W

德军密码

德军密码

100

- 二战时盟军截获德军一段密码，密文：00000110000000001010101101110010110001011000001110011001100111100111001（密钥：helloworld），你可能会解出一个keyxxxxx的答案，请在y后面加{，结尾加}，答案的格式是key{xxxxx}，所以答案是

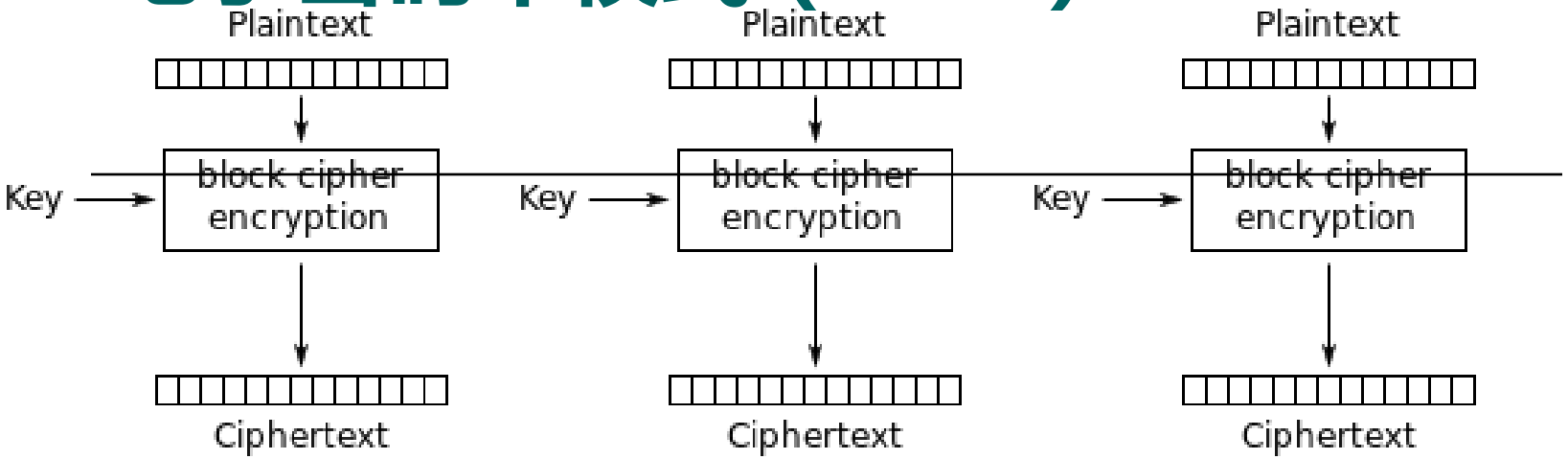
2.6 分组密码工作模式

- **对称分组密码一次处理一个数据块.**
  - **在DES和3DES的情况下，块长度是b = 64位.**
  - **对于AES，块长度为b = 128**
  - **对于较长的明文量，有必要将明文分解为b位块，必要时填充最后一个块.**
- **NIST已经定义了五种操作模式.**
  - **旨在涵盖几乎所有可能使用分组密码的加密应用程序.**
  - **适用于任何对称分组密码，包括三重DES和AES**

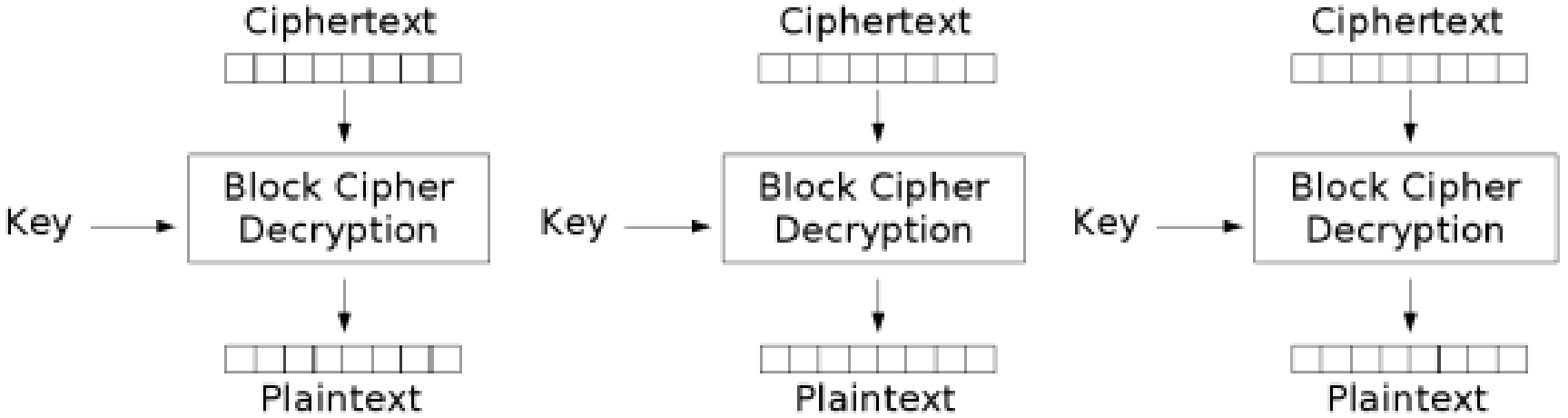
2.6.1 电子密码本模式（ECB）

- 明文每次处理b比特，并且使用相同的密钥对每个明文块进行加密.
- 使用术语“密码本”是因为对于给定的密钥，每个b比特的明文块都有一个唯一的密文。
  - 可以想象一个巨大的码本，其中每个可能的b位明文模式都有一个条目，显示其对应的密文.
- 对于ECB，如果在消息中出现相同的b位明文块，则它始终生成相同的密文。
  - 因此，对于冗长的消息，ECB模式可能不安全。
  - 如果消息是高度结构化的，那么密码分析者可能会利用这些规则。

电子密码本模式（ECB）



Electronic Codebook (ECB) mode encryption



Electronic Codebook (ECB) mode decryption

XOREncode之ECB示例

2.6.2 密文分组链接模式（CBC）

- **CBC**模式中，加密算法的输入是当前明文分组与前一密文分组的异或；每个分组使用同一密钥。加密函数的每次输入和明文之间的关系不固定。因此，**64**比特的重复模式并不会被暴露。
- 加密：

$$C_j = E(K, [C_{j-1} \oplus P_j])$$

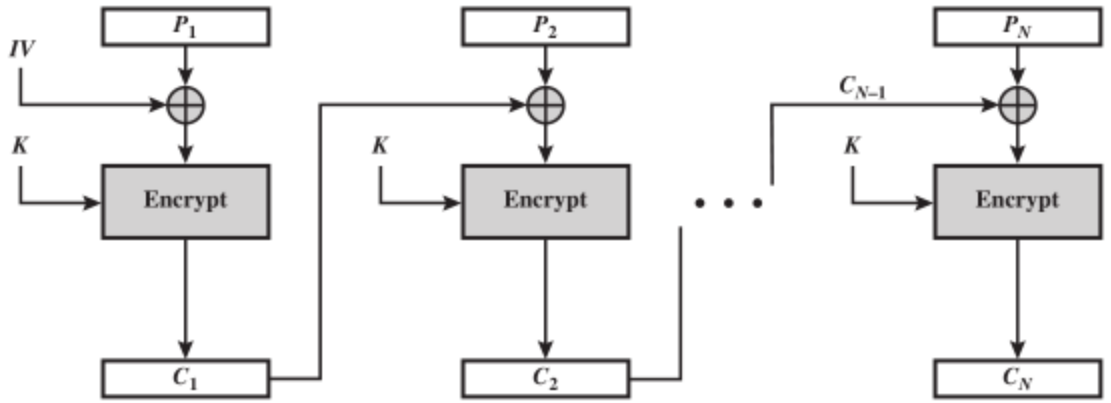
- 解密：

$$D(K, C_j) = D(K, E(K, [C_{j-1} \oplus P_j]))$$

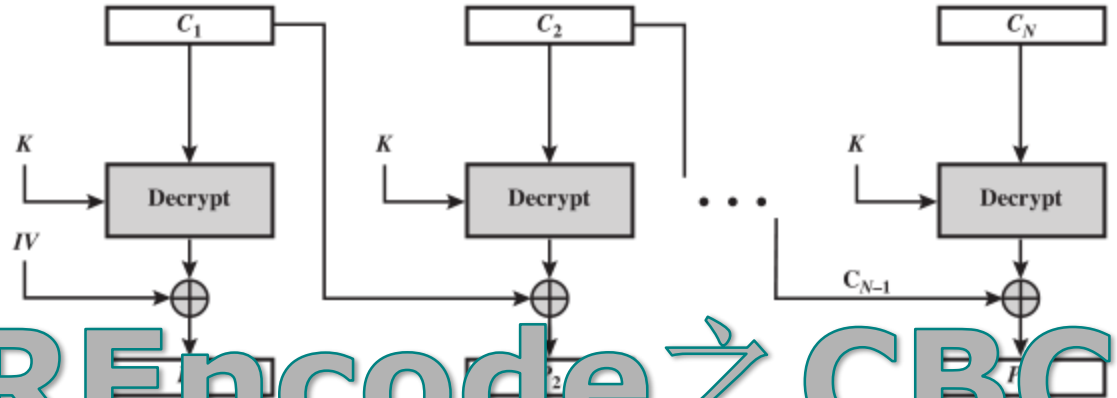
$$D(K, C_j) = C_{j-1} \oplus P_j$$

$$C_{j-1} \oplus D(K, C_j) = C_{j-1} \oplus C_{j-1} \oplus P_j = P_j$$

图2.9 密文分组链接（CBC）模式



(a) Encryption



(b) Decryption

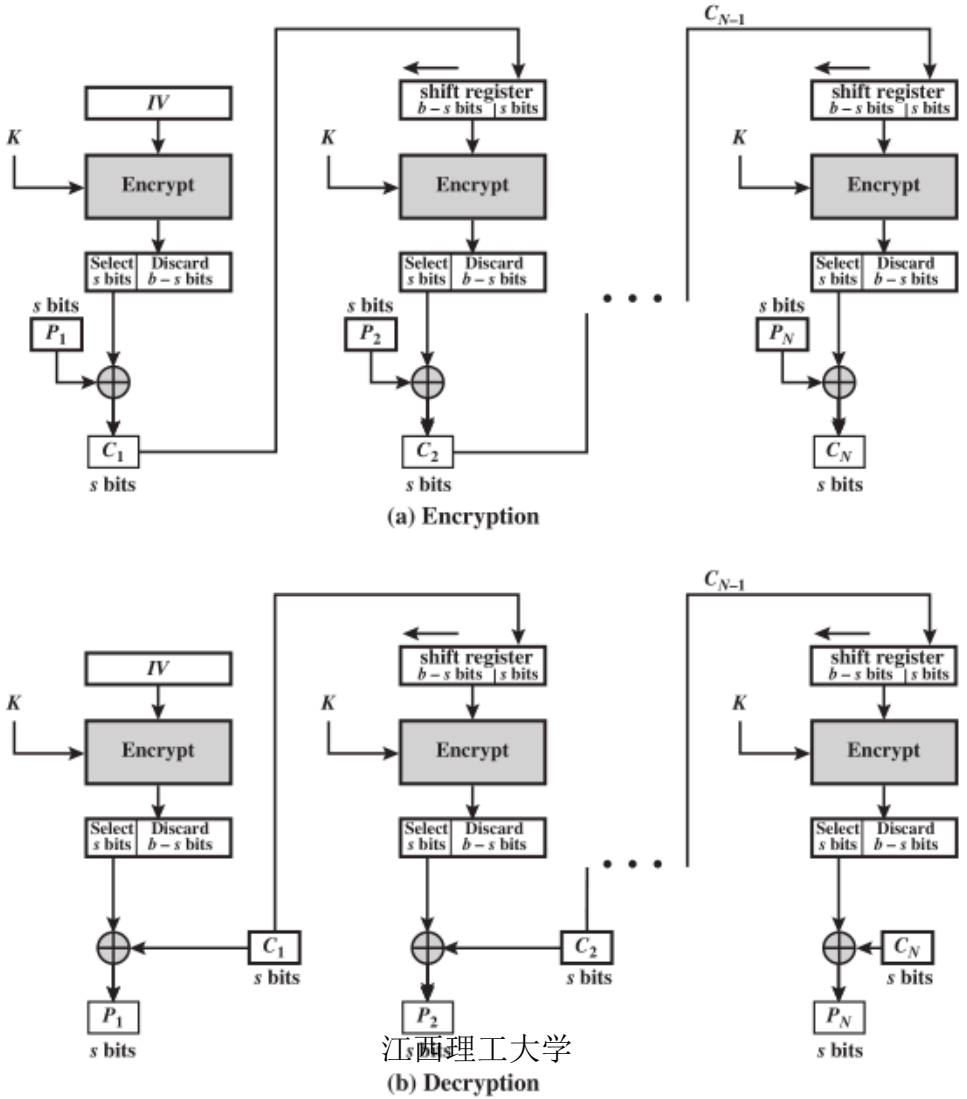
XOREncode之CBC示例

2.6.3 密文反馈模式（CFB）

- **使用密码反馈（CFB）模式能将任意分组密码转化为流密码。**

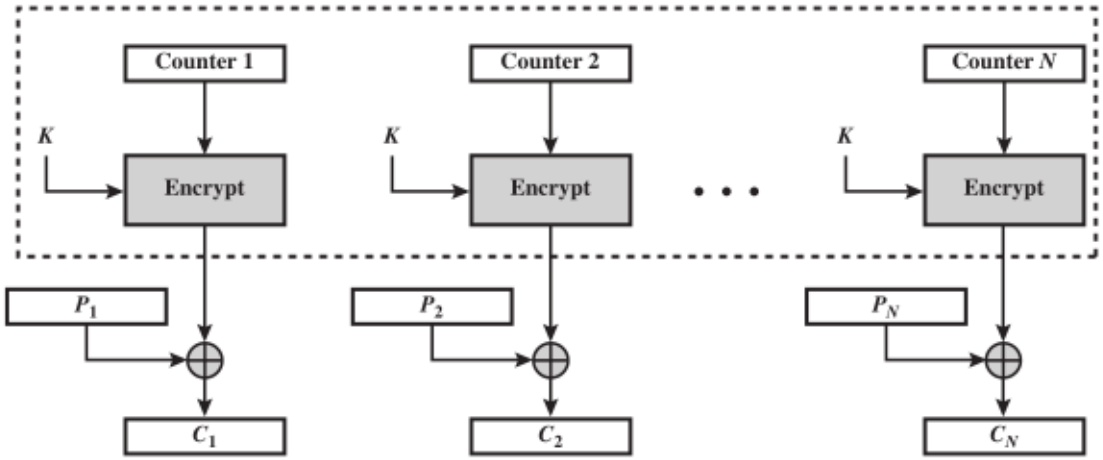
加密模块的输入是一个64比特的移位寄存器，初始值设定为某一初始向量（IV）。加密模块输出最左边（最高）s比特和明文P1的第1个单元进行异或，产生密文C1的第1个单元，然后传输。接下来，移位寄存器的内容都左移s比特，同时将C1放在移位寄存器的最右边（最低）s比特。

图2.10 密文反馈模式（CFB）模式

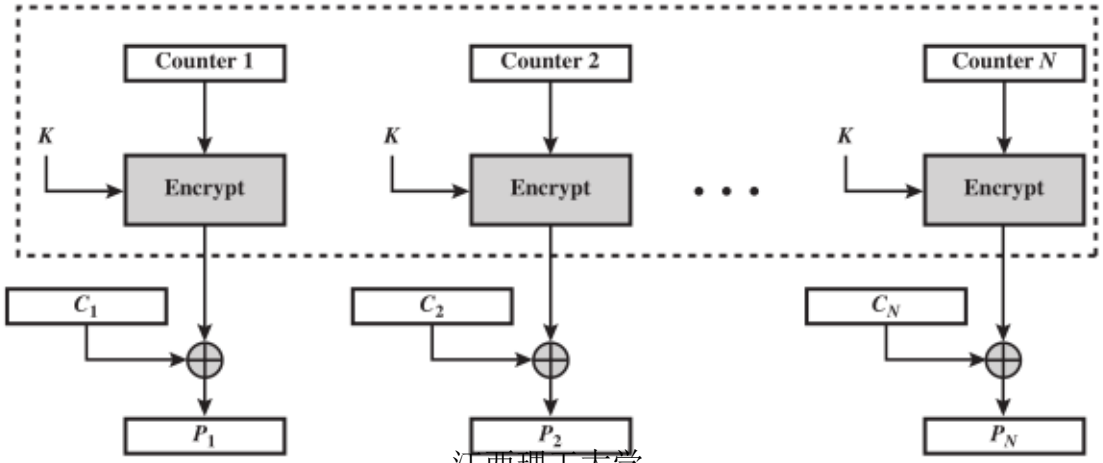


移位寄存器

2.6.4 计算器模式（CTR）



(a) Encryption



(b) Decryption

CTR模式的优点

- **硬件效率**
  - 加密/解密可以在多个明文或密文块上并行完成.
  - 吞吐量仅受实现的并行数量的限制。
- **软件效率**
  - 由于并行执行的机会，可以有效地利用支持并行功能的处理器.
- **预处理**
  - 底层加密算法的执行不依赖于明文或密文的输入。 当呈现明文或密文输入时，唯一的计算是一系列XOR，大大提高了吞吐量。

CTR模式的优点（续）

- **随机访问**
  - 可以以随机访问方式处理明文或密文的第i个块。
- **可证明的安全性**
  - 可以证明，CTR至少与本节中讨论的其他模式一样安全.
- **简单性**
  - 仅需要实施加密算法而不是解密算法。

小结

- 古典密码学
- 对称密码学
- RC4流密码