

密码学基础

@密码的基本概念

@古典密码

@隐写术与其它

@代换技术

@置换技术

@对称密钥密码算法

@公开密钥密码算法

两种加密形式

@广泛使用的两种加密形式是 传统加密和公钥加密

@传统加密又称为对称加密、单钥加密

@代换密码、置换密码、二者组合

@缺陷：安全性在于保持算法本身的保密性

@不适合大规模生产

@不适合较大的或者人员变动较大的组织

@用户无法了解算法的安全性

基本概念

@明文(plain text)：原始可以理解的信息或者数据。

@密文(cipher text)：加密后的信息。

@加密(encrypt, encryption)：从明文到密文的变换过程。

@解密(decrypt, decryption)：从密文到明文的变换过程。

@密码算法(Algorithm)：用来加密/解密的数学函数

@ $c=E(m)$, $m=D(c)$, $D(E(m))=m$ ；也简称为密码(Cipher)

@密钥(Key)：密码算法中的一个变量

@ $c=E_{Ke}(m)$, $m=D_{Kd}(c)$, $D_{Kd}(E_{Ke}(m))=m$

密码的基本概念

两种加密形式

@现代加密又称为非对称加密、公钥加密，于1976年第一次公开发表

@把算法和密钥分开

@密码算法可以公开，密钥保密

@密码系统的安全性在于保持密钥的保密性

@适于大规模生产

基本概念

@密码编码学(Cryptography)

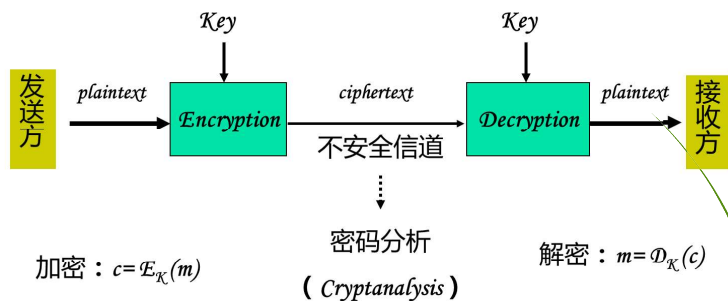
@研究各种加密方案的学科。

@密码分析学(Cryptanalysis)

@研究破译密码获得消息的学科，又称“破译”。

@密码编码学和密码分析学统称为密码学。

密码学的基本模型



密码编码学

@密码编码学的特征一：转换明文为密文的运算类型

- ②所有的加密算法基于两个原理：
置换(Transposition)和代换(Substitution)
- ②置换是将明文中的元素重新排列
- ②代换是将明文中每个元素映射成为另外一个元素
- ②原则是不允许丢失信息，即所有的运算都是可逆的

密码分析学

@攻击传统的密码体制有两种方法： 密码分析学和穷举攻击

@密码分析学

- ②利用算法的特征来推导出特别的明文或者使用的密钥
- ②如果能够成功地推导出密钥，那么影响将是灾难性的

密码编码学

@研究各种加密方案的学科称为密码编码学

@密码编码学系统具有三个独立的特征：

- ②转换明文为密文的运算类型
- ②所用的密钥数
- ②处理明文的方法

密码编码学

@密码编码学的特征二：所用的密钥数

- ②发送方和接收方使用相同的密钥，称为对称密码、传统加密
- ②发送方和接收方使用不相同的密钥，称为非对称密码、公钥密码

@密码编码学的特征三：处理明文的方法

- ②分组密码/块密码(Block Cipher):
每次处理一个输入分组，相应地输出一个输出分组
- ②流密码/序列密码(Stream Cipher):
连续地处理输入元素，每次输出一个元素

密码分析学

@穷举攻击

- ②对一条密文尝试所有可能的密钥，直至密文可以转化为可读的有意义的明文
- ②一般，要获得成功必须尝试所有可能密钥的一半

Key Size (bits)	Number of Alternative Keys	Time required at 10^6 Decryption/ μ s
32	$2^{32} = 4.3 \times 10^9$	2.15 milliseconds
56	$2^{56} = 7.2 \times 10^{16}$	10 hours
128	$2^{128} = 3.4 \times 10^{38}$	5.4×10^{18} years
168	$2^{168} = 3.7 \times 10^{50}$	5.9×10^{30} years

密码分析学

@根据已知信息量的多少，可以将攻击方法分成如下几类：

- @唯密文攻击(Ciphertext only)
- @已知明文攻击(Known plaintext)
- @选择明文攻击(Chosen Plaintext)
- @选择密文攻击(Chosen Ciphertext)
- @选择密钥攻击(Chosen CipherKey)

无条件安全和计算安全

@无论有多少可以使用密文，都不足以唯一地确定由该体制产生密文所对应的明文，则加密机制是无条件安全的

@一种加密机制在计算上是安全的，当它满足以下条件之一：
(computationally secure)

- @破译密码的代价大于加密数据本身的价值
- @破译密码的时间超过了密文信息的生命期

异或举例



@异或

异或运算（不带进位加法）：

明文:	0	0	1	1	
加密:	密钥:	0	1	0	1
	密文:	0	1	1	0

$$C = P \oplus K$$

密文:	0	1	1	0	
解密:	密钥:	0	1	0	1
	明文:	0	0	1	1

$$P = C \oplus K$$

$$K = C \oplus P$$

- @已知明文、密文，怎样求得密钥？
- @只知道密文，如何求得明文和密钥？

密码分析

@Success in dealing with unknown ciphers is measured by these four things in the order named, **perseverance, careful methods of analysis, intuition, luck**.
The ability at least to read the language of the original text is very desirable but not essential.

@Parker Hitt “军事密码破译指南”的开场白

一个简单的加密算法—异或

$$0 \oplus 0 = 0 \quad x \oplus 0 = x$$

$$1 \oplus 1 = 0 \quad x \oplus x = 0$$

$$1 \oplus 0 = 1 \quad x \oplus 1 = x^{-1}$$

$$0 \oplus 1 = 1 \quad x \oplus x^{-1} = 1$$

古典密码 之 隐写术及其它

诗情画意传“密语”

早妆未罢暗凝眉， 芦花丛中一扁舟，
迎户愁看紫燕飞， 俊杰俄从此地游，
无力回天春已老， 义士若能知此理，
双栖画栋不如归。 反躬难逃可无忧。

隐写术及其它



@隐写术是隐藏信息的存在

@常见的方法有：

②字符标记、不可见墨水、针刺、打字机的色带校正

@优点：

②能够被某些人使用而不容易被发现

@缺点：

②形式简单但构造费时，要求有大量的开销来隐藏相对较少的信息

②一旦构造方法被发现，就会变得完全没有价值

②隐写术一般没有稳健性

代换技术Substitution Cipher

@代换技术是将明文字母替换成其他字母、数字或者符号的方法

②Caesar密码

②单表代换密码

②Playfair密码

②Hill密码

②Vigenere密码

②Vernam密码和一次一密

诗情画意传“密语”

王先生：

来信收悉，你的盛情真是难以报答。我已在昨天抵达广州。秋雨连绵，每天需备伞一把方能上街，苦矣。大约本月中旬我才能返回，届时再见。

王先生：

来信收悉，你的盛情真是难以报答。我已在昨天抵达广州。秋雨连绵，每天需备伞一把方能上街，苦矣。大约本月中旬我才能返回，届时再见。

古典密码之代换技术

Caesar 密码

@最早的代换密码是由Julius Caesar发明的；原理是对字母表中的每个字母用它之后的第3个字母代换

@Caesar 密码： $c = (m + 3) \bmod 26$

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
密文	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

@可以采用穷举攻击破译Caesar 密码，因为

②已知Caesar 密码的加密和解密算法

②只需要测试25个密钥

②明文所用的语言是已知的，意义易于识别

关键词密码

- ① 设一个关键词放在前面，其余字母按顺序排列

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
密文	T	S	I	N	G	H	U	A	B	C	D	E	F	J	K	L	M	O	P	Q	R	V	W	X	Y	Z

- ② Caesar密码和关键词密码都是单表代换密码

- ③ 密文是26个字母的任意置换，即从明文字母到密文字母的映射
- ④ 密文中还是带有原始字母使用频率的一些统计学特征，比较容易被攻破

Playfair密码

- ① Playfair密码是最著名的多表代换密码，它把明文中的双字母音节作为一个单元并将其转换成密文的双字母音节。Playfair算法基于一个由关键词构成的5×5字母矩阵

- ② 举例：关键词是monarchy

- ③ 填充矩阵的方法是关键词从左至右、从上至下填在矩阵格子里，再将剩余的字母按照字母表的顺序从左至右、从上至下填在剩余的格子里。字母I和J当作一个字母

M	O	N	A	R
C	H	Y	B	D
E	F	G	I/J	K
L	P	Q	S	T
U	V	W	X	Z

Playfair密码

- ① Playfair密码相对于简单的单表密码是巨大的进步
- ② 因为26×26个字母对，比单字母判定要困难得多
- ③ 在很长一段时间内，Playfair密码被认为是牢不可破的，并且在一战和二战中被大量使用
- ④ 但是Playfair密码依然完整地保留了明文语言的结构，几百个字母密文就足够分析出规律了

多表代换密码

- ① 代换密码必须考虑的一个问题是：明文的语法模式和结构有多少仍然保留在密文中

- ② 减少这种保留信息的方法有：

- ③ 对明文中的多个字母一起加密
- ④ 采用多表代换密码

Playfair密码

- ① 对明文按照如下规则一次加密两个字母
- ② 如果该字母对是两个相同的字母，添加一个填充字

- ③ 例如x，先把balloon变成 ba lx lo on 四个字母对

- ④ 落在矩阵同一行的明文字母对中的字母，由其右边的字母代换。

- ⑤ 例如：ar被代换为RM

- ⑥ 落在矩阵同一列的明文字母对中的字母由其下面的字母来代换

- ⑦ 例如：mu被代换为CM

- ⑧ 其它的明文字母对对字母按照如下方法代换：它所在的行是该字母所在的行，列是另外一个字母所在的列

- ⑨ 例如：hs被代换为BP

M	O	N	A	R
C	H	Y	B	D
E	F	G	I/J	K
L	P	Q	S	T
U	V	W	X	Z

Hill密码

- ① Hill密码是1929年由数学家Lester Hill发明的一个多表代换密码
- ② 密码算法是将m个连续的明文字母代换成m个密文字母。这是由m个线性等式决定的，在等式里每个字母被指定为一个数字（a=0, b=1, ..., z=25）。
- ③ 例如：m=3，系统可以描述为：

- $c_1 = (k_{11}p_1 + k_{12}p_2 + k_{13}p_3) \bmod 26$
- $c_2 = (k_{21}p_1 + k_{22}p_2 + k_{23}p_3) \bmod 26$
- $c_3 = (k_{31}p_1 + k_{32}p_2 + k_{33}p_3) \bmod 26$

Hill密码

@用矩阵表示 $C=KP \bmod 26$

@C是密文, K是 3×3 的加密密钥信, P是明文

$$\begin{bmatrix} C_1 \\ C_2 \\ C_3 \end{bmatrix} = \begin{bmatrix} k_{11} & k_{12} & k_{13} \\ k_{21} & k_{22} & k_{23} \\ k_{31} & k_{32} & k_{33} \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix} \bmod 26$$

@例: 对明文 “paymoremoney”用密钥K加密

$$K = \begin{pmatrix} 17 & 17 & 5 \\ 21 & 18 & 21 \\ 2 & 2 & 19 \end{pmatrix} \quad \begin{aligned} C &= K(p a y) \bmod 26 \\ &= K(15 \ 0 \ 24) \bmod 26 \\ &= (375 \ 819 \ 486) \bmod 26 \\ &= (11 \ 13 \ 18) = LNS \end{aligned}$$

Hill密码练习

@例: Hill密码的秘钥K如下所示, 请对明文 “THU” 加密

$$K = \begin{pmatrix} 17 & 17 & 5 \\ 21 & 18 & 21 \\ 2 & 2 & 19 \end{pmatrix}$$

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
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

$$@C=K(THU) \bmod 26$$

$$@C=K(19 \ 7 \ 20) \bmod 26$$

$$@C=(542 \ 945 \ 432) \bmod 26$$

$$@C=(22 \ 9 \ 16)=WJQ$$

Vernam密码和一次一密

@1918年, 工程师Gilbert Vernam引入了这样一种体制: 选择一个与明文毫无统计关系并且和明文一样长的密钥, 其运算是基于二进制数据而非字母的

$$@ci = pi \oplus ki$$

@陆军情报局军官Joseph Mauborgne又提出了一种Vernam密码的改进方案:

@使用与消息一样长且无重复的随机密钥来加密消息, 这就是著名的一次一密, 它是不可攻破的, 即无条件安全的

Hill密码

@Hill密码解密需要用到K矩阵的逆 K^{-1} 。

@ $K^{-1} K = K K^{-1} = I$ (单位矩阵)

@并不一定所有的矩阵都有逆。

@加密过程 $C=KP \bmod 26$

@解密过程 $P= K^{-1} C \bmod 26$

@Hill的优点是完全隐蔽了单字母的频率特性。Hill用的矩阵越大, 所隐藏的信息就越多

@例如 3×3 的矩阵就隐藏了双字母的频率特性

Vigenere密码

@多表代换密码的共同特性:

@采用相关的单表代换规则集

@由密钥决定给定变换的具体规则

@Vigenere密码的代换规则由26个类Caesar密码代换表构成, 其中每个代换表是对明文字母表移位0到25次后得到的代换单表

@每个密码代换表由一个密钥字母表示, 这个密钥字母就是用来代换明文字母a的

@Vigenere密码的强度在于每个明文字母对应着多个密文字母, 并且使用唯一的字母作为密钥词。因此字母频率信息被隐藏了, 但并非所有的明文结构信息都被隐藏了

Vernam密码和一次一密



@一次一密的安全性完全取决于密钥的随机性。但实际上一次一密很少使用, 因为:

@产生大规模随机密钥很困难

@密钥的分配和保护很困难

古典密码 之 置换技术

置换技术举例

@明文:

@Attack postponed until two am

@一次置换后密文:

@TTNAAPMTSUOAODWCOIX
KNLYPETZ

@二次置换后密文:

@NSCYAUOPTTWLTMDNAOIE
PAXTOKZ

@经过多次置换后，已没有什么
规律可寻了

密钥	4	3	1	2	5	6	7
明文	a	t	t	a	c	k	p
	o	s	t	p	o	n	e
	d	u	n	t	i	l	t
	w	o	a	m	x	y	z

密钥	4	3	1	2	5	6	7
明文	t	t	n	a	a	p	t
	m	t	s	u	o	a	o
	d	w	c	o	i	x	k
	n	l	y	p	e	t	z

转轮机

@英国的TYPEX打字密码机，是德国3轮ENIGMA的改进型密码机；它在英国通信中使用广泛，且在破译密钥后帮助破解德国信号



置换技术Transposition Cipher

@置换密码是通过置换形成新的排列

@最简单的例子是栅栏技术

@按照对角线的顺序写入明文，按照行的顺序读出做为密文

@更复杂的方案是把消息一行一行写成矩形块，然后按列读出，同时把列的次序打乱。列的次序就是算法的密钥

@单步置换还是容易被识破，一般采用多步置换密码就安全多了

转轮机

@转轮机采用多层加密原理。

包括一组相互独立的旋转轮，电脉冲可以通过。每个圆筒有26个输入引脚和26个输出引脚，并且一一相连

@每个圆筒就定义了一个单表代换，上一个圆筒的输出连接下一个圆筒的输入，多个圆筒就是多表代换

@转轮密码机ENIGMA，由Arthur Scherbius于1919年发明，4轮ENIGMA在1944年装备德国海军

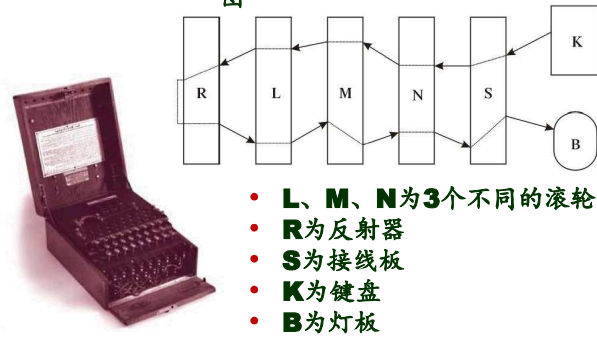


SPY Museum



Enigma密码机加密原理

Enigma M3型密码编码路径图



Enigma：滚轮和反射器的设置



N	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	D	F	H	J	L	C	P	R	T	X	V	Z	N	Y	E	I	W	G	A	K	M	U	S	Q	O
N1	D	F	H	J	L	C	P	R	T	X	V	Z	N	Y	E	I	W	G	A	K	M	U	S	Q	O	B
N2	F	H	J	L	C	P	R	T	X	V	Z	N	Y	E	I	W	G	A	K	M	U	S	Q	O	B	D
N3	H	J	L	C	P	R	T	X	V	Z	N	Y	E	I	W	G	A	K	M	U	S	Q	O	B	D	F

M	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
	A	J	D	K	S	I	R	U	X	B	L	H	W	T	M	C	Q	G	Z	N	P	Y	F	V	O	E

L	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
	E	K	M	F	L	G	D	Q	V	Z	N	T	O	W	Y	H	X	U	S	P	A	I	B	R	C	J

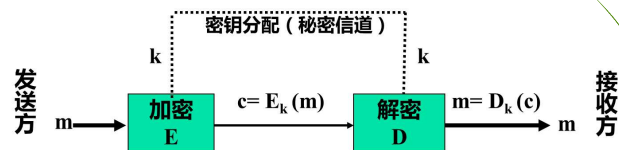
R	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
	Y	R	U	H	Q	S	L	D	P	X	N	G	O	K	M	I	E	B	F	Z	C	W	V	J	A	T

对称密钥密码算法

- @密码的基本概念
- @古典密码
- @对称密钥密码算法
 - @对称密码简介
 - @简化DES算法
 - @简化DES实例
 - @Feistel密码结构
 - @DES密码算法
 - @常用的对称密码
- @公开密钥密码算法

对称密钥算法简介

- @加密和解密使用相同的密钥: $KE=KD$
- @密钥必须使用秘密的信道分配



简化DES算法

对称密钥密码简介

对称密钥算法简介



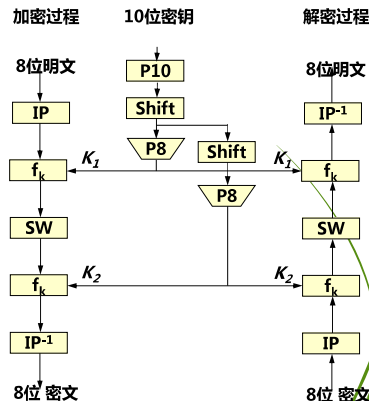
- @常用对称密钥密码算法
 - @DES (Data Encryption Standard)及其各种变形
 - @IDEA (International Data Encryption Algorithm)
 - @RC2, RC4, RC5,
 - @AES (Advanced Encryption Standard)
 - @CAST-128
 - @Blowfish

简化DES

- @简化DES(S-DES)是为教学使用的一个加密算法，与DES有着相似的性质和结构，但是参数要小很多，便于理解。
- @S-DES加密算法的输入为一个8位明文组和一个10位的密钥，输出为8位的密文组。
- @S-DES解密算法的输入为一个8位密文组和一个10位的密钥，输出为8位的明文组。

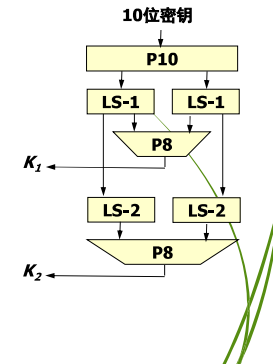
- @加密算法包括5个函数：
- @初始置换IP
- @复杂函数fk，包含了置换和代换运算，并且依赖于密钥
- @用于转换数据两个部分的简单置换函数SW
- @再一次的复杂函数fk
- @置换函数IP的逆函数IP-1
- @密文= IP-1 (fk2 (SW (fk1 (IP(明文))))))
- @明文= IP-1 (fk1 (SW (fk2 (IP(密文))))))
- @K1=P8(Shift(P10(key)))
- @K2=P8(Shift(Shift(P10(key))))

S-DES的整体结构

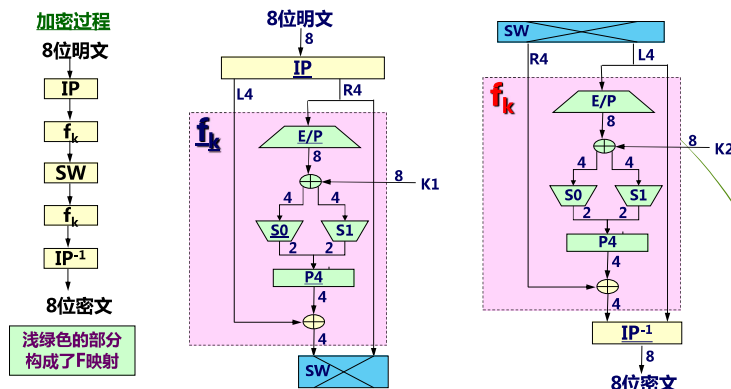


S-DES密钥的产生

- @S-DES依赖于收发双方共享的10位密钥，它产生的两个8位子密钥分别用在加密和解密的不同阶段。
- @10位密钥即(k1,k2,k3,k4,k5,k6,k7,k8,k9,k10)。
- @置换P10定义为P10(k1,k2,k3,k4,k5,k6,k7,k8,k9,k10)=(k3,k5,k2,k7,k4,k10,k1,k9,k8,k6)
- @LS-1表示前5位和后5位分别循环左移1位。
- @LS-2表示前5位和后5位分别循环左移2位。
- @置换P8定义为P8(k1,k2,k3,k4,k5,k6,k7,k8,k9,k10)=(k6,k3,k7,k4,k8,k5,k10,k9)
- @由此可以计算得到子密钥K1和K2。



S-DES加密过程



S-DES加密

- @加密过程包括5个函数
- @初始置换IP
 - @IP(n1,n2,n3,n4,n5,n6,n7,n8)=(n2,n6,n3,n1,n4,n8,n5,n7)
- @E/P(n1,n2,n3,n4)=(n4,n1,n2,n3,n2,n3,n4,n1)
- @将8位子密钥K1或K2与之进行按位异或。
- @前面4位(第一行)输入到S盒S0中得到一个2位输出，后面4位(第二行)输入到S盒S1中得到另外一个2位输出。

S-DES加密：映射F

- @S盒的操作如下：
 - @第1、4位作为二进制数决定S盒的行
 - @第2、3位作为二进制数决定S盒的列
 - @输出即是二进制的2位输出
- @例如:对S0盒输入(0100)
 - @计算行(P1,P4)=(00);计算列 (P2,P3)=(10)
 - @输出是S0盒第0行第2列的数3，即二进制 (11)。

$$S_0 = \begin{matrix} & 0 & 1 & 2 & 3 \\ \begin{matrix} 0 \\ 1 \\ 2 \\ 3 \end{matrix} & \begin{bmatrix} 1 & 0 & 3 & 2 \\ 3 & 2 & 1 & 0 \\ 0 & 2 & 1 & 3 \\ 3 & 1 & 3 & 2 \end{bmatrix} \end{matrix}$$

$$S_1 = \begin{matrix} & 0 & 1 & 2 & 3 \\ \begin{matrix} 0 \\ 1 \\ 2 \\ 3 \end{matrix} & \begin{bmatrix} 0 & 1 & 2 & 3 \\ 1 & 2 & 0 & 1 & 3 \\ 2 & 3 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 & 3 \end{bmatrix} \end{matrix}$$

S-DES加密：映射F

- @由S0和S1输出构成的4位再进行一次置换
 - P4(n1,n2,n3,n4)=(n2,n4,n3,n1)
- @P4输出再和L4按位异或，和R4组合在一起就是函数F的输出
- @交换函数SW
 - @SW将输入的左4位和右4位交换
- @末尾置换IP-1
 - @IP-1(n1,n2,n3,n4,n5,n6,n7,n8)=(n4,n1,n3,n5,n7,n2,n8,n6)

S-DES加密：复杂函数fk



@复杂函数fk

@设L和R分别是fk的8位输入的左边4位和右边4位。F是一个4位串到4位串的映射。则

$$fk(L,R)=(L\oplus F(R,SK),R)$$

@其中SK是子密钥。

@例：IP置换的输出为(1011 1101)， $F(1101,SK) = (1110)$ ，求fk。
则 $L\oplus F(R,SK) = (1011)\oplus(1110) = (0101)$ ，所以fk (1011 1101)=(0101 1101)。

S-DES实例

@使用S-DES，

用密钥(01 11 11 11 01)手工加密二进制串 (1010 0010)

$$P_{10}(k_1,k_2,k_3,k_4,k_5,k_6,k_7,k_8,k_9,k_{10})= (k_3,k_5,k_2,k_7,k_4,k_{10},k_1,k_9,k_8,k_6)$$

$$P_8(k_1,k_2,k_3,k_4,k_5,k_6,k_7,k_8,k_9,k_{10})= (k_6,k_3,k_7,k_4,k_8,k_5,k_{10},k_9)$$

$$IP(n_1,n_2,n_3,n_4,n_5,n_6,n_7,n_8)= (n_2,n_6,n_3,n_1,n_4,n_8,n_5,n_7)$$

$$IP^{-1}(n_1,n_2,n_3,n_4,n_5,n_6,n_7,n_8)= (n_4,n_1,n_3,n_5,n_7,n_2,n_8,n_6)$$

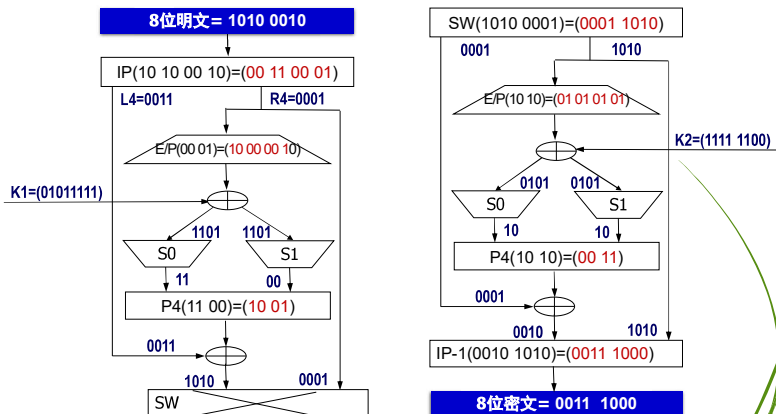
$$E/P(n_1,n_2,n_3,n_4)=(n_4,n_1,n_2,n_3,n_2,n_3,n_4,n_1)$$

$$P_4(n_1,n_2,n_3,n_4)=(n_2,n_4,n_3,n_1)$$

$$S_0 = \begin{matrix} & 0 & 1 & 2 & 3 \\ 0 & 1 & 0 & 3 & 2 \\ 1 & 3 & 2 & 1 & 0 \\ 2 & 0 & 2 & 1 & 3 \\ 3 & 3 & 1 & 3 & 2 \end{matrix}$$

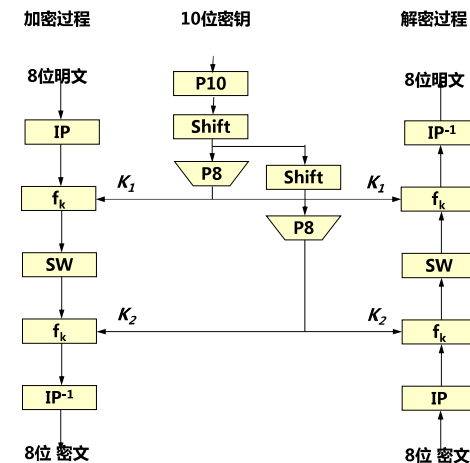
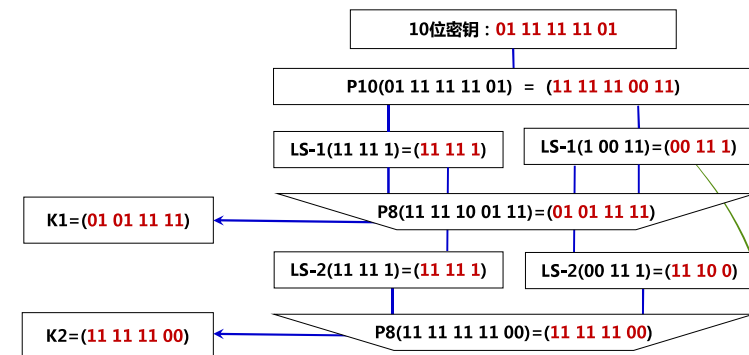
$$S_1 = \begin{matrix} & 0 & 1 & 2 & 3 \\ 0 & 0 & 1 & 2 & 3 \\ 1 & 2 & 0 & 1 & 3 \\ 2 & 3 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 & 3 \end{matrix}$$

S-DES实例：加密过程



简化DES实例

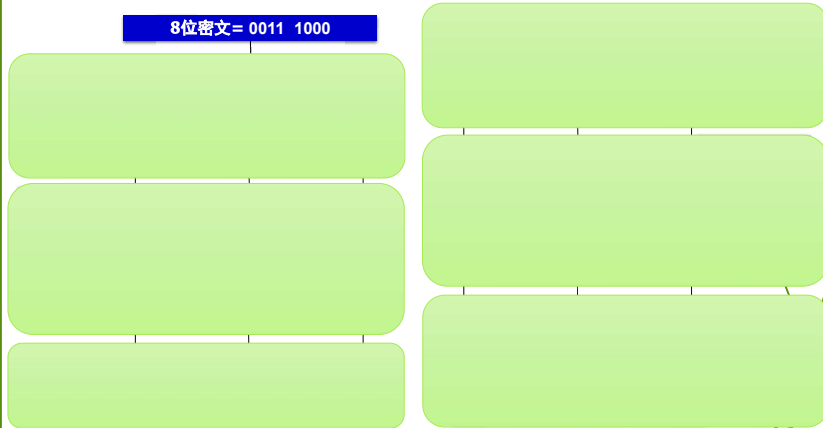
S-DES实例：计算子密钥



**S-DES的
整体结构**

S-DES实例：解密过程

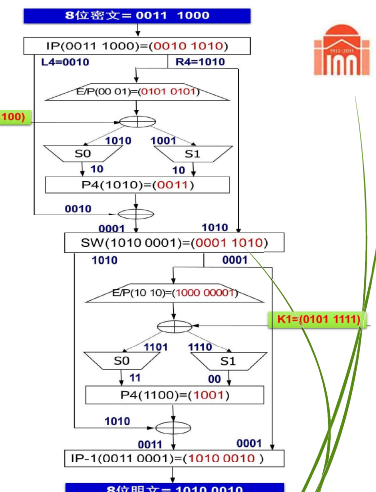
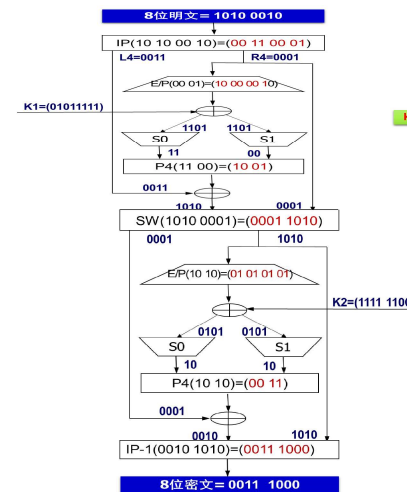
8位密文 = 0011 1000



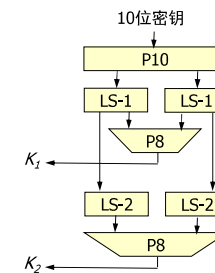
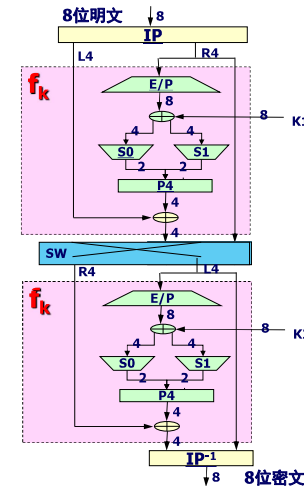
Feistel密码结构

流密码和分组密码

- ④流密码每次加密数据流的一位或一个字
 - ④Vigenere密码和Vernan密码都是流密码
- ④分组密码是将一个明文组作为整体加密并且得到的是与之等长的密文组
 - ④典型的分组大小是64bits或者128bits
 - ④绝大多数基于网络的对称密码应用使用的都是分组密码
- ④现在使用的对称分组密码算法都基于Feistel分组密码结构的



S-DES密码算法



Feistel Cipher Structure

- ④Feistel建议使用乘积密码的概念来逼近简单代换密码
- ④乘积密码是指依次使用两个或以上的基本密码，所得密码强度将强于所有单个密码的强度
- ④Feistel 建议交替使用代换和置换
 - ④这实际上是Claude Shannon提出的交替使用混淆(Confusion)和扩散(Diffusion)乘积密码的实际应用
- ④几乎所有的传统的分组密码算法的结构都有和 Feistel Cipher类似的结构

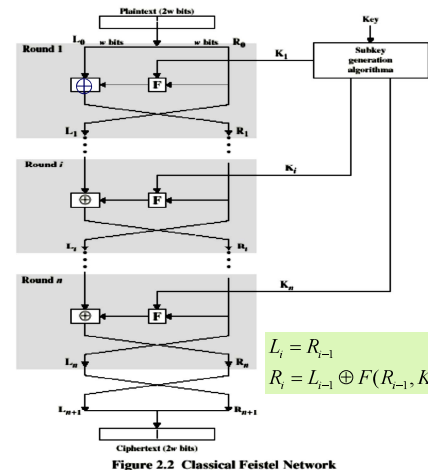
混淆和扩散

- @Shannon引入混淆和扩展来刻画任何密码系统的两个基本构件，所关注的是如何挫败基于统计方法密码分析
- @**扩散**是指使明文的统计特征消散在密文中，让每个明文数字尽可能地影响多个密文数字
- @**混淆**是尽可能地使密文和加密密钥间的统计关系更复杂，以挫败推导出密钥的企图
- @混淆和扩散正是抓住了设计分组密码的本质而成为现代分组密码设计的里程碑

Feistel密码结构

- @每轮迭代都有相同的结构
 - ③代换作用在数据左半部分，它通过轮函数F作用在数据的右半部分后，与左半部分进行异或来完成
 - ③每轮迭代的轮函数相同，但是输入的子密钥 K_i 不同
 - ③代换之后，交换数据的左右两半完成置换

DES密码算法



Feistel密码结构

- @输入是长度为 $2w$ 位的明文组和密钥 K
- @明文组被分成两个部分： L_0 和 R_0 ；这两半数据经过 n 轮迭代后组合成密文组
- @第 i 轮迭代的输入来自于上轮迭代的输出
- @子密钥 K_i 是由整个密钥 K 推导出来的
 - ③ K_i 不同于 K ，也互不相同

影响Feistel具体实现的重要参数



- @分组长度和密钥长度
 - ③越长安全性越高，但是会降低加解密速度
- @迭代轮数
 - ③多轮加密可以取得高安全性
- @子密钥产生算法
 - ③子密钥产生越复杂，密码分析攻击就越困难
- @轮函数
 - ③轮函数越复杂，抗攻击能力越强
- @在实现中还需要考虑快速软件加解密和简化分析难度等问题

DES的诞生

- @20世纪60年代后期，IBM公司成立了一个由Horst Feistel负责的计算机密码编码学研究项目
- @1971年设计出算法LUCIFER：分组长度为64位，密钥长度为128位，具有Feistel结构的分组密码
- @因为LUCIFER非常成功，IBM决定开发一个适合于芯片实现的商业密码产品
- @由Walter Tuchman和Carl Meyer负责，开发了一个抗分析攻击能力更强，密钥长度为56位的密码，这就是DES
- @1977年，DES被美国国家标准局接收为国家密码标准方案

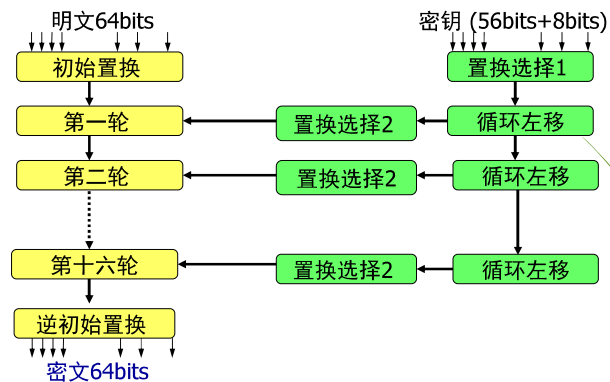
DES算法原理

@DES采用分组加密，是一种对称密钥算法

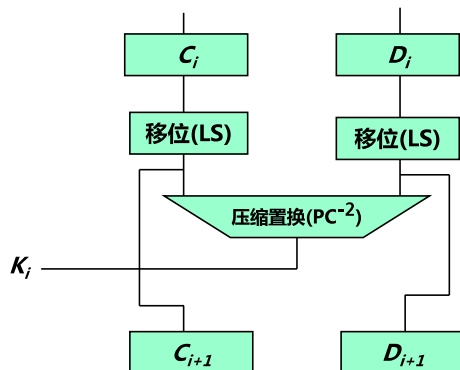
- @密钥长度为56bits
(加上奇偶校验，通常写成64bits)
- @分组长度是64 bits

@DES算法使用了标准的算术和逻辑运算

DES算法概要



子密钥生成



DES 加密过程

@首先把明文分成以64 bit为单位的块m，对于每个m，执行如下操作

$$\text{DES}(m) = \text{IP}^{-1} \cdot \text{T}_{16} \cdot \text{T}_{15} \cdot \dots \cdot \text{T}_2 \cdot \text{T}_1 \cdot \text{IP}(m)$$

- @初始置换，IP
- @16轮迭代， T_i , $i=1,2,\dots,16$
- @末尾置换， IP^{-1}

@除了初始置换和末尾置换，DES的结构与Feistel密码结构完全相同

子密钥生成

@拆分：

- @56 bits 的密钥分成两部分， C_i , D_i ，各28bits

@循环左移：

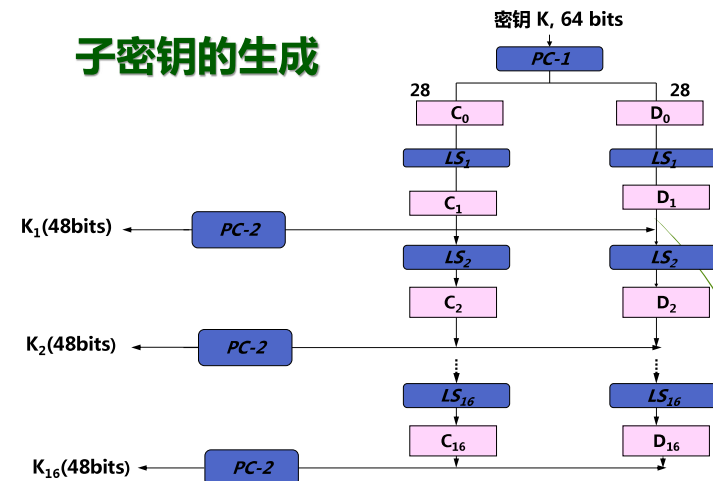
- @根据迭代的轮数，分别左移一位或两位

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

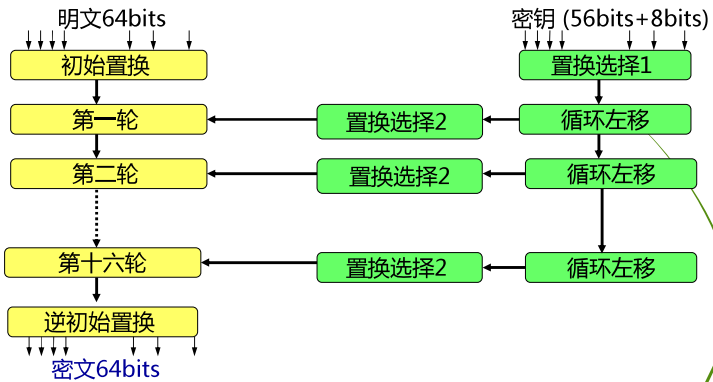
@压缩置换（PC-2）：从56bits中选择48bits

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

子密钥的生成



DES算法概要



末尾置换 (IP-1)

@末置换

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

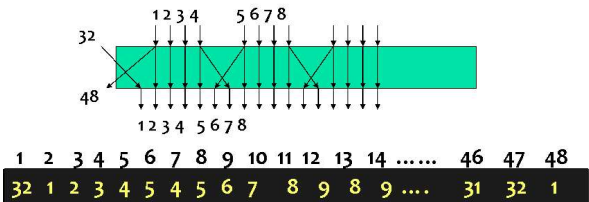
@IP-1(IP(M))=M

E盒扩展置换

@将Ri从32位扩展到48位

@目的:

输入的一位影响下一步的两个替换，使得输出对输入的依赖性传播得更快，密文的每一位都依赖于明文的每一位



初始置换 (IP)

@初始换位 (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

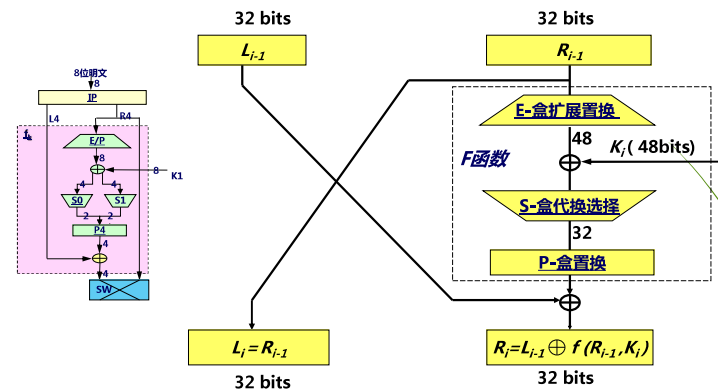
$$\mathbf{M} = m_1 \ m_2 \ \dots m_{62} \ m_{63} \ m_{64}$$

IP(M)

$$M' = m_{58} m_{50} \dots m_{23} m_{15} m_7$$

一轮迭代

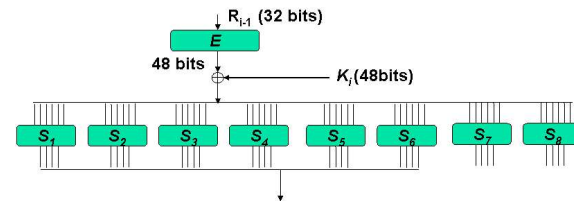
$$\begin{aligned} L_i &= R_{i-1} \\ R_i &= L_{i-1} \oplus F(R_{i-1}, K_i) \end{aligned}$$



S-盒代换选择

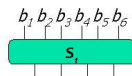
④将48比特压缩成32比特。代换函数由8个S盒组成，每个S盒都有6位输入产生4位输出。

@盒Si输入(abcdef), 其中(a)决定了Si盒中的行, (bcde)决定了Si盒中列, 取得Si盒中一个元素N, 将N转换成4位二进制即得到Si盒输出。



S-盒代换选择

@输入6比特: $b_1b_2b_3b_4b_5b_6$

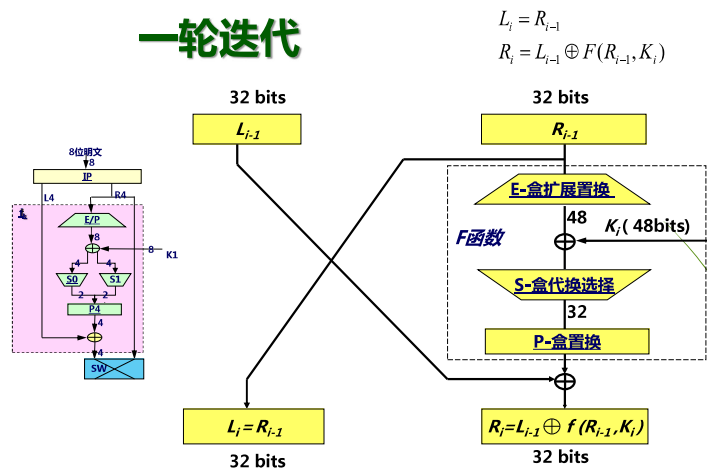


@输出4比特: $S(b_1b_6, b_2b_3b_4b_5)$

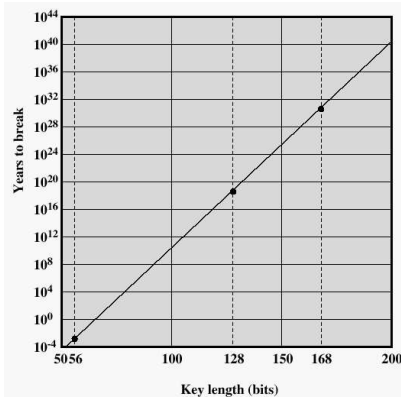
@举例: $S_1(100110) = S_1(10, 0011) = S_1(2, 3) = 8 = (1000)$

		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
S1	0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
	1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
	2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
	3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13
S2	0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
	1	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
	2	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
	3	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.

一轮迭代



DES的安全性和速度



@密钥的长度与安全性

@1976年, 耗资2000万美元的计算机, 可以在一天中找到密钥。

@1993年, 100万美元的计算机, 3.5小时用穷举法找到密钥。

@现在, 5000\$的计算机, 可以在3.5小时找到密钥

P-盒置换

@P-盒的输出: 输入32bits, 输出32bits。

1	2	3	4	5	6	7	8	9											30	31	32										
16	7	20	21	29	12	28	17	1	15	.	.	..	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.

DES解密过程

@DES解密过程与加密过程完全相似, 只不过将16次迭代的子密钥顺序倒过来, 即

$$m = DES^{-1}(c) = IP^{-1} \cdot T_1 \cdot T_2 \cdot \dots \cdot T_{15} \cdot T_{16} \cdot IP(c)$$

@可以证明:

$$DES(DES^{-1}(m)) = m$$

软硬件实现的速度



@硬件实现: 商业DES芯片

@VLSI 公司 VM009 1993年制造 200M Bytes/s

@软件实现:

@80486

@CPU 66Hz, 每秒加密43000个DES分组, 336K Bytes/s

@HP 9000/887

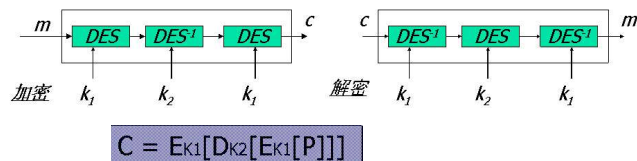
@CPU 125 Hz, 每秒加密196,000个分组, 1.53M Bytes/s

常用的对称密码

- @三重DES算法
- @Blowfish算法
- @RC5密码算法
- @AES密码算法

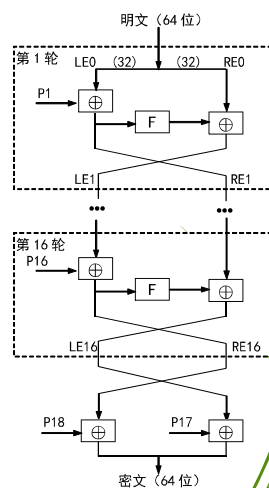
3 DES

- @三重DES加密，密钥长度为112比特。已知明文攻击的代价上升到了 2^{112} 数量级，远远超出了人类现行的和未来的能力
- @两个密钥的三重DES是目前被广泛接收的加密算法，密钥 $k=k_1k_2$



Blowfish加密过程

- @Blowfish算法使用两个基本的运算：
 - @加法：字的模223加法(+)
 - @按位异或：($\oplus$)
- @F映射中包含了四个S盒运算



对称密码

- @当前最重要的几种对称密码算法，3DES、Blowfish和RC5，都具有以下几个标准：
 - @有相当高的密码强度
 - @广泛应用于Internet上面
 - @代表了自DES以来的对称密码技术
- @高级加密标准AES是一个对称分组密码算法，是美国国家标准技术局NIST2001年发布的，目的是用来取代DES。NIST预计在可以预见的未来3DES仍将是一个被认可的算法

Blowfish算法

- @Blowfish算法是由Bruce Schneier设计的一种对称分组密码；具有快速、紧凑、简单和安全性可变的特性
- @Blowfish算法的分组长度是64位，密钥长度可以在32~448位之间变化
- @Blowfish算法在很多产品上得到实现，其安全性至今没有受到挑战

Blowfish算法

- @Blowfish算法的子密钥和S盒都是用Blowfish算法本身生成的，这使得数据完全不可以辨认，对它的密钥分析也就异常困难
- @Blowfish算法每轮运算都是对数据的左右两个部分同时执行运算，与古典Feistel结构不同。这使得密码的强度又增强了
- @因为密钥长度可以达到448位，完全可以抵抗穷举攻击

RC5

@RC5是Ron Rivest设计的一种对称加密算法。

RC5适用于软硬件实现、快速、迭代次数和密钥长度可变、简单、安全性高

@RC5的分组长度可以位32、64或者128位，密钥长度则可取0~2040位

@RC5是一个由三个参数确定的加密算法族

@w: 分组长度

@r: 迭代次数

@b: 密钥K的8位字节数

AES的产生

@如果仅仅考虑算法安全，3DES能够称为未来数十年的加密标准。但是，3DES最根本的缺点在于该算法用软件实现的速度太慢。另外，DES和3DES的分组长度都是64位，还应该再长一些，比较安全

@NIST在1997年公开征集新的高级加密标准AES，高求安全性能不低于3DES，同时具有更好的可执行性。而且提出，AES必须是分组长度为128位的对称分组密码，并能够支持128位、192位和256位的密钥

AES的结构

@AES不是Feistel结构

@AES的每一轮都使用代换和置换并行的处理整个数据分组

@该结构由四个阶段构成，包括一个置换和三个代换；而且每个阶段均可逆

@字节代换：用一个S盒完成分组中的字节代换

@行移位：一个简单的置换

@列混淆：算术代换

@轮密钥加：利用当前分组和扩展密钥的一部分进行按位异或

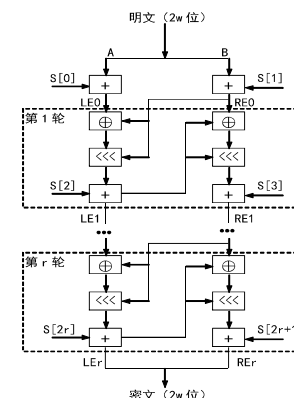
RC5加密算法

@RC5使用了下面三种运算：

@加法：
字的模 2^w 加法(+)

@按位异或： $\oplus$

@循环左移：
字x循环左移(<<<)



AES产生

@2001年，NIST从15个方案中选取了Rijndael作为AES算法。Rijndael的作者是比利时的密码学家Joan Daemen博士和Vincent Rijmen博士

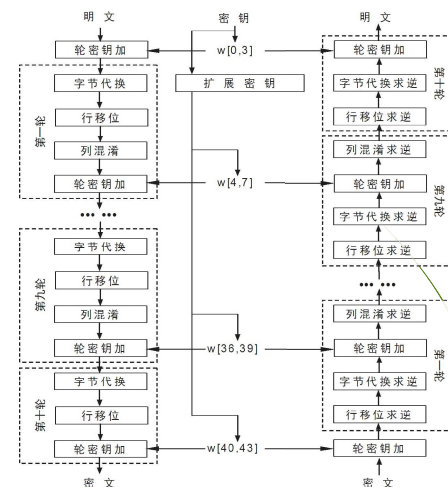
@在AES中，分组长度128位，密钥长度128位。并具有如下特征：

@对所有已知的攻击具有免疫力

@在各种平台上，其执行速度快并且代码紧凑

@设计简单

AES的加密和解密



AES的结构

@算法结构简单

③对于加密和解密，算法从轮密钥加开始，接着执行 九轮迭代，每轮包含四个阶段，最后执行包含三个阶段的最后一轮

@密钥被扩展成44个32位字所构成的数组w[i]

③在每轮的加密解密过程中，有四个32位字的密钥作为该轮的轮密钥

@仅仅在轮密钥加阶段使用密钥；轮密钥加实际上是一种 Vernam密码。轮密钥加和混淆等交替使用，是非常有效和安全的

其他对称密码算法



算法	密钥长度	迭代次数	数学操作	应用
DES	56	16	XOR, S-Box	Kerberos, SET
3DES	112 or 168	48	XOR, S-Box	PGP, S/MIME
IDEA	128	8	XOR, +, ×	PGP
Blow Fish	最大448	16	XOR, S-Box, +	
RC5	最大2048	<255	+, -, XOR	
CAST-128	40—128	16	+, -, S-Box	PGP

公开密钥密码算法

- @密码的基本概念
- @古典密码
- @对称密钥密码算法
- @公开密钥密码算法
 - @非对称密码算法原理
 - @数论基础
 - @RSA算法
 - @DH密钥交换算法

2

非对称密码算法

- @对称密钥密码系统的缺陷
 - @密钥必须经过安全的信道分配
 - @无法用于数字签名
 - @密钥管理复杂，密钥的数量： $O(n^2)$
- @1976年，Whitfield Diffie和Martin Hellman提出了非对称密钥密码（一个公钥、一个私钥），也称公钥密码
- @公钥密码是密码学历史上唯一的一次真正的革命，它是基于数学函数而不是代换和置换

4

公钥密码体制

- @公钥密码体制有6个组成部分。
 - @明文：可读的信息，做为加密算法的输入。
 - @加密算法：对明文进行的各种变换。
 - @公钥/私钥：一个用于加密，一个用于解密。加密算法执行的变换依赖于公钥和私钥。
 - @密文：加密算法的输出，不可读信息。密文依赖于明文和密钥，不同的密钥产生不同的密文。
 - @解密算法：根据密文和相应的密钥，产生出明文。

6

非对称密码算法原理

公钥密码和之前四千多年来所使用的所有密码学方法都不同，是密码学中的一个惊人的成就！

3

5

符号说明

- @会话密钥 K_s
- @用户A的公钥 K_{Ua}
- @用户A的私钥 K_{Ra}
- @ $E_{K_{Ua}}[P]$ ：用A的公钥对明文P加密
- @ $E_{K_{Ra}}[P]$ ：用A的私钥对明文P加密
- @ $D_{K_{Ua}}[C]$ ：用A的公钥对密文C解密
- @ $D_{K_{Ra}}[C]$ ：用A的私钥对密文C解密

7

公钥密码体制

@每个用户产生一对密钥：用于加密和解密。其中一个密钥存于公开的寄存器或者文件中，即公钥；另外一个密钥是私有的，称为私钥

@公钥公开，用于加密和验证签名

@私钥保密，用作解密和签名

@例如：

@Bob给Alice发消息，用Alice的公钥对信息加密

@Alice收到消息后，用自己的私钥解密

@由于只有Alice拥有自己的私钥，所以其他接收者都不能解密出其信息来

8

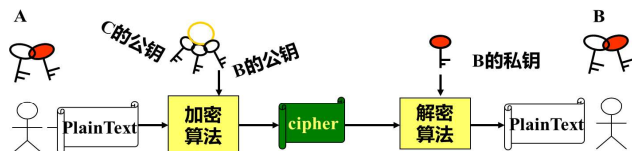
公钥密码系统的加密原理

@每个通信实体有一对密钥（公钥,私钥）。公钥公开，用于加密和验证签名；私钥保密，用作解密和签名

@A向B发送消息，用B的公钥加密

@B收到密文后，用自己的私钥解密

@任何人向B发信息都使用同一个密钥(B的公钥)加密。没有人可以得到B的私钥，只有B可以解密



10

公钥密码算法的表示

@对称密钥算法

@密钥：会话密钥(K_s)

@加密函数： $C = E_{K_s}[P]$

@对密文C，解密函数： $D_{K_s}[C]$

@公开密钥算法

@A(K_{Ua} , K_{Ra})向B(K_{Ub} , K_{Rb})发送信息：

@加密： $C = E_{K_{Ub}}[P]$ (用B的公开密钥加密)

@解密： $P = D_{K_{Rb}}[C]$

@签名： $E_{K_{Ra}}[P]$ (用A的私有密钥加密)

@验证： $D_{K_{Ua}}[C]$

12

公钥密码体制

@利用这种方法，通信各方皆可以访问公钥，而私钥是个通信方在本地产生的，所以不必进行分配

@只要系统控制了私钥，那么它的通信是安全的

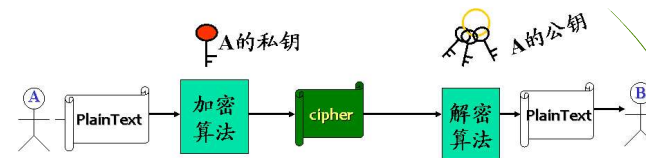
@任何时刻，系统都可以改变自己的私钥，而公布相应的公钥代替原来的公钥

9

公钥密码系统的签名原理

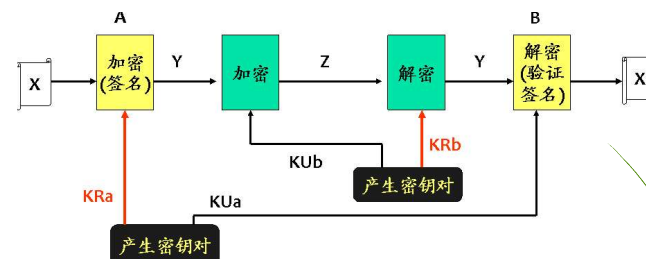
@A向B发送消息，用A的私钥加密(签名)

@B收到密文后，用A的公钥解密(验证)



11

数字签名和加密同时使用



$$Y = E_{K_{Ra}}(X), Z = E_{K_{Ub}}[Y] = E_{K_{Ub}}[E_{K_{Ra}}(X)]$$
$$Y = D_{K_{Rb}}(Z), X = D_{K_{Ua}}[Y] = D_{K_{Ua}}[D_{K_{Rb}}(Z)]$$

13

公钥密码的数学原理：陷门单向函数

- @公钥密码系统是基于陷门单向函数的概念
- @单向函数是求逆困难的函数；单向陷门函数，是在不知陷门信息下求逆困难的函数，当知道陷门信息后，求逆是易于实现的
 - @单向陷门函数 $f(x)$ ，必须满足以下三个条件：
 - ①给定 x ，计算 $y=f(x)$ 是容易的
 - ②给定 y ，计算 x 使 $y=f(x)$ 是困难的
(所谓计算 $x=f^{-1}(y)$ 困难是指计算上相当复杂已无实际意义)
 - ③存在 δ ，已知 δ 时对给定的任何 y ，若相应的 x 存在，则计算 x 使 $y=f(x)$ 是容易的
- @仅满足①、②两条的为单向函数；
第③条为陷门性， δ 称为陷门信息

14

公钥密码分析

- @与对称密码相同，公钥密码也容易受到穷举攻击
 - @攻击者可以利用公钥对所以可能的密钥加密，并与所传送的密文对比，从而可以解密任何消息
 - @公钥体制使用的是某种可逆的数学函数，函数值的复杂性可能不是密钥长度的线性增长，而是指数或者更快的增长速度
- @对应穷举攻击，解决方法也是使用长密钥
 - @为了抗击穷举攻击，密钥必须足够长，为了便于实现，密钥又要足够短
- @在实际应用中，公钥密码目前仅局限于
密钥管理和数字签名

16

对称密码 vs 非对称密码

- @误解二：公钥密码是一种通用方法，传统密码已经过时
 - @由于公钥密码需要大量计算，仅限于密钥管理和签名这类应用中，所以基本不太可能取代传统密码
- @误解三：传统密码中与密钥分配中心的握手是一件异常麻烦的事情，而公钥密码实现密钥分配则是非常简单的
 - @使用公钥密码也需要某种形式的协议，通常也包括一个中心代理，所包含的处理过程即不比传统密码简单，也不比其更有效

18

公钥密码系统的应用

- @公钥密码系统有三种用途：
 - @加密/解密
 - @数字签名：如果电子文件都需要签名，如何能够确保数字签名是出自某个特定人，而且通信双方无异议
 - @发送方用自己的**私钥**签署报文，接收方用对方的公钥验证对方的签名
 - @密钥交换：双方协商会话密钥，用于对称密钥数据加密

公钥密码算法	加密/解密	数字签名	密钥交换
RSA	Y	Y	Y
Diffie-Hellman	N	N	Y
DSA	N	Y	N

15

对称密码 vs 非对称密码

- @与只使用一个密钥的对称传统密码不同，公钥密码学是非对称的，使用两个独立的密钥，这对于密钥分配、数字签名、认证技术等有很深远影响。但是现在对非对称密码有很多误解
- @误解一：从密码分析角度看，公钥密码比传统密码更安全
 - @任何加密方法的安全性依赖于密钥长度和破译密文所需要的计算量。从抗击密码分析的角度看，不能简单地认为传统密码和公钥密码那个更安全

17

对称密码和公钥密码



	对称密码	公钥密码
一般要求	加密和解密使用相同的密钥	加密和解密使用相同的密码算法，但使用不同的密钥
	收发双方必须共享密钥	发送方拥有加密或解密的密钥，而接收方只拥有另一个密钥。
安全要求	密钥必须是保密的	两个密钥之一必须是保密的
	若没有其它信息，则解密消息是不可能或者至少是不可行的	若没有其他信息，则解密消息是不可能或者至少是不可行的。
	知晓算法和若干密文不足以确定密钥	知晓算法和其中一个密钥以及若干密文不足以确定另外一个密钥

19

数论基础

20

数论基础：模运算

@模运算 mod

@ $a = qn + r$ 其中 $0 \leq r < n$; $q = [a/n]$;
则有 $r = a \bmod n$

@ $[x]$ 表示小于或等于x的最大整数

@如果 $(a \bmod n) = (b \bmod n)$,
则称a与b模n同余,
记为 $a \equiv b \pmod n$

@例如, $2^3 \equiv 8 \pmod 5$, $8 \equiv 1 \pmod 7$

22

数论基础：欧拉函数

@欧拉函数 $\phi(n)$: n 是正整数, $\phi(n)$ 是比 n 小且与 n 互素的正整数个数

@ $\phi(3) = \{1, 2\} = 2$

@ $\phi(4) = \{1, 3\} = 2$

@ $\phi(5) = \{1, 2, 3, 4\} = 4$

@ $\phi(6) = \{1, 5\} = 2$

@ $\phi(7) = \{1, 2, 3, 4, 5, 6\} = 6$

@ $\phi(10) = \{1, 3, 7, 9\} = 4$

@如果 p 是素数, 则 $\phi(p) = (p-1)$

@如果 p, q 是素数, 则
 $\phi(pq) = \phi(p) \phi(q) = (p-1)(q-1)$

24

数论基础：素数

@素数: 整数 $P > 1$ 是素数, 当且仅当它只有因子+1和+ P .

@100以内的素数

@2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79
83 89 97

@设 P 是所有素数的集合, 则任意正整数可以唯一地表示为:

@a与b的最大公因数: $\gcd(a, b)$

@ $\gcd(20, 24) = 4$, $\gcd(15, 16) = 1$

@如果 $\gcd(a, b) = 1$, 称a与b互素

21

数论基础：模运算

@模运算对加法和乘法是可交换、可结合、可分配的

@ $(a+b) \bmod n = ((a \bmod n) + (b \bmod n)) \bmod n$

@ $(a-b) \bmod n = ((a \bmod n) - (b \bmod n)) \bmod n$

@ $(a \times b) \bmod n = ((a \bmod n) \times (b \bmod n)) \bmod n$

@ $(a \times (b+c)) \bmod n =$
 $((a \times b) \bmod n + (a \times c) \bmod n) \bmod n$

@幂, 模运算 $ma \bmod n$

@ $m^2 \bmod n =$
 $(m \times m) \bmod n = (m \bmod n)^2 \bmod n$

@ $m^4 \bmod n = (m^2 \bmod n)^2 \bmod n$

@ $m^8 \bmod n = ((m^2 \bmod n)^2 \bmod n)^2 \bmod n$

@ $m^{25} \bmod n = (m \times m^8 \times m^{16}) \bmod n$

23

数论基础：欧拉定理



@欧拉定理

@若整数 m 和 n 互素, 则 $m^{\phi(n)} \equiv 1 \pmod n$

等价形式 $m^{\phi(n)+1} \equiv m \pmod n$

@例如:

@ $m=3, n=10$; $\phi(10)=4$; $m^{\phi(n)}=3^4=81$; $81 \bmod 10 = 1$

@即: $81 \equiv 1 \pmod 10$; $3^{4+1} = 243 \equiv 3 \pmod 10$

@推论:

@给定两个素数 p, q , $p < q$, 两个整数 n, m , 使得 $n=pq$, $0 < m < n$,
则对于任意整数 k , 下列关系成立: $m^{k\phi(n)+1} \equiv m \pmod n$

25

RSA算法

26

RSA算法：密钥的产生

@密钥产生

- ①取两个大素数 p, q , 保密;
- ②计算 $n=pq$, 公开 n ;
- ③计算欧拉函数 $\phi(n) = (p-1)(q-1)$;
- ④任意取一个与 $\phi(n)$ 互素的小整数 e , 即 $\gcd(e, \phi(n))=1$; $1 < e < \phi(n)$
- ⑤寻找 d , $d < \phi(n)$, 使得 $de \equiv 1 \pmod{\phi(n)}$, 即 $de = k\phi(n) + 1$
- ⑥公开 (e, n)
- ⑦将 d 保密, 丢弃 p, q .

@公开密钥: $KU = \{e, n\}$

@秘密密钥: $KR = \{d, n\}$

- ① 设: $p=7, q=17$
- ② 则: $n=119$
- ③ $\Phi(n) = 6 \times 16 = 96$
- ④ 选择 $e=5$
- ⑤ $5d = k \times 96 + 1$
- ⑥ 令 $k=4$, 得到 $d=77$
- ⑦ 故知道:
 - ① $KU = \{5, 119\}$
 - ② $KR = \{77, 119\}$

28

RSA 算法的证明

@试证明：解密过程是正确的

@证明：

$$\begin{aligned} M &= C^d \pmod{n} \\ &= (M^e \pmod{n})^d \pmod{n} \\ &= M^{ed} \pmod{n} \\ \text{即 } M^{ed} &\equiv M \pmod{n} \end{aligned}$$

@根据欧拉定理推论: $M^{k\phi(n)+1} \equiv M \pmod{n}$, 得到 $ed = k\phi(n) + 1$

30

RSA算法简介

@MIT的Ron Rivest, Adi Shamir, Leonard Adleman于1977年提出, 于1978年首次发表的RSA算法, 是最早满足要求的、被广泛接收、而且被实现的通用公钥密码算法之一

@RSA体制是一种分组密码, 其明文和密文都是 $0 \sim n-1$ 之间的整数, 通常 n 的大小为1024位二进制数或者309位十进制数



27

RSA 算法：加密/解密

@利用：公钥 $KU = \{e, n\}$ 和私钥 $KR = \{d, n\}$

@加密过程

@把待加密的内容分成 k 比特的分组, $k \leq \log_2 n$, 并写成数字, 设为 M ,
则: $C = M^e \pmod{n}$

@例如: $C = M^5 \pmod{119}$

@解密过程

@ $M = C^d \pmod{n}$

@例如: $M = C^{77} \pmod{119}$

29

RSA加密过程举例

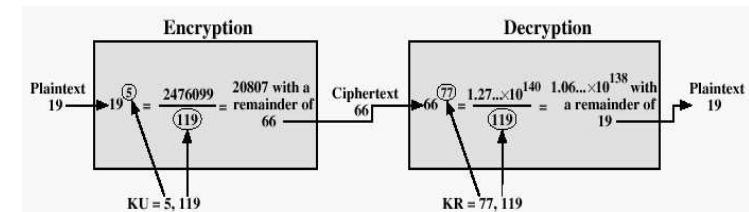


Figure 3.9 Example of RSA Algorithm

31

@用RSA算法对下面数据实现加密和解密。

@ $p=5$; $q=11$; $e=3$; $M=9$

@密钥产生

@取两个大素数 p, q , 保密;

@计算 $n=pq$, 公开 n ;

@计算欧拉函数
 $\phi(n)=(p-1)(q-1)$;

@任意取一个与 $\phi(n)$ 互素的小整数 e ,
即 $\gcd(e, \phi(n))=1$; $1 < e < \phi(n)$

@寻找 d , $d < \phi(n)$, 使得
 $de \equiv 1 \pmod{\phi(n)}$,
即 $de = k\phi(n) + 1$

@公开 (e, n)

@将 d 保密, 丢弃 p, q .

@公开密钥: $KU=(e, n)$

@秘密密钥: $KR=(d, n)$

RSA算法举例1

• 设: $p=5, q=11$
• 则: $n=5 \times 11=55$
• $\Phi(n)=(5-1) \times (11-1)=4 \times 10=40$

• 因为 $e=3$
根据 $de = k\phi(n) + 1$
故得 $3d = k \times 40 + 1$

• 令 $k=2$, 得到 $d=27$

• 故知道:
• $KU = \{3, 55\}$
• $KR = \{27, 55\}$

• 加密过程
 $C = M^e \pmod n$
 $= 9^3 \pmod{55} = 729 \pmod{55} = 14$

• 解密过程
 $M = C^d \pmod n$
 $= 14^{27} \pmod{55} = 9$

32

RSA算法举例2

@第二步: 明文信息数字化

@假定明文英文字母编码表为按字母顺序排列数值

@则得到分组后的key的明文信息为: 11, 05, 25。

字母	A	B	C	D	E	F	G	H	I	J	K	L	M
码值	01	02	03	04	05	06	07	08	09	10	11	12	13
字母	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
码值	14	15	16	17	18	19	20	21	22	23	24	25	26

34

RSA 算法的安全性

@对RSA算法的攻击方法: 蛮力攻击、数学攻击、计时攻击

@蛮力攻击: 对所有密钥都进行尝试

@数学攻击: 有多种数学攻击方法, 他们的实质是两个素数乘积(n)的因子分解

@计时攻击: 类似于通过观察他人转动保险柜拨号盘的时间长短来猜测密码; 攻击者可以通过记录计算机解密消息所用的时间来确定私钥

@计时攻击不仅可以用于攻击RSA, 还可以用于攻击其它公钥密码系统。由于这种攻击的完全不可预知性以及它仅仅依赖明文, 所以计时攻击具有很大的威胁

36

RSA算法举例2

@假设Alice需要将明文“key”通过RSA加密后传递给Bob。

@第一步: 设计公钥和密钥

@令 $p=3$, $q=11$, 得出 $n=p \times q=3 \times 11=33$

@ $\Phi(n)=(p-1)(q-1)=2 \times 10=20$

@取 $e=3$, 则 $e \times d \equiv 1 \pmod{\Phi(n)}$, 即 $3 \times d \equiv 1 \pmod{20}$; 取 $d=7$

@从而可以设计出一对公私密钥,

@加密密钥(公钥)为: $KU=(e, n)=(3, 33)$

@解密密钥(私钥)为: $KR=(d, n)=(7, 33)$

33

RSA算法举例2

@第三步: 明文加密

@Alice用加密密钥(3,33)将数字化明文分组信息加密成密文

@由 $C \equiv M^e \pmod n$ 得到

相应的密文信息为:

11, 26, 16

$C1 \equiv (M1)^e \pmod n = 11^3 \pmod{33} = 11$
 $C2 \equiv (M2)^e \pmod n = 5^3 \pmod{33} = 26$
 $C3 \equiv (M3)^e \pmod n = 25^3 \pmod{33} = 16$

@第四步: 密文解密

@Bob收到密文后, 若将其解密, 只需要计算 $m_i = C_i^d \pmod n$

@ $M_1=11^7 \pmod{33}=11$; $M_2=26^7 \pmod{33}=05$; $M_3=16^7 \pmod{33}=25$

@得到明文信息为: 11, 05, 25; 根据编码表将其转换为“key”

35

RSA 算法的安全性

@大数因子分解是数论中的一个难题, 因子分解的进展可用MIPS年描述计算的代价

@MIPS年是指一台每秒执行百万条指令的处理器运行一年

十进制数字位数	近似比特数	完成日期	MIPS年
100	332	1991年4月	7
110	365	1992年4月	75
120	398	1993年6月	830
129	428	1994年4月	5000
130	431	1996年4月	1000
140	465	1999年2月	2000
155	512	1999年8月	8000

37

RSA 算法的性能



@与其它密码体制一样，RSA抗穷举攻击的方法也是使用大密钥空间，但是密钥越大，系统运行速度越慢

- @软件实现比DES慢100倍
- @硬件实现比DES慢1000倍

	512位	768位	1024位
加密	0.03	0.05	0.08
解密	0.16	0.48	0.93
签名	0.16	0.52	0.97
验证	0.02	0.07	0.08

38

Diffie-Hellman密钥交换算法

@1976年，Diffie-Hellman第一个发表的公开密钥算法，被称为Diffie-Hellman密钥交换算法

@Diffie-Hellman密钥交换算法的目的是使两个用户能够安全地交换密钥，该算法本身也只局限于进行密钥交换

@Diffie-Hellman密钥交换算法的有效性在于计算离散对数非常困难

40

离散对数

@对于整数 b ($b < p$) 和素数 p 的一个本原根 a ，可以找到一个唯一的指数 i ，使得：

$$b \equiv a^i \pmod{p}, \text{ 其中 } 0 \leq i \leq (p-1)$$

i 称为 b 的以 a 为底模 p 的离散对数或指数，记为 $\text{ind}_{a,p}(b)$

@ $\text{ind}_{a,p}(1)=0$ ，因为 $a^0 \pmod{p} = 1 \pmod{p} = 1$;

@ $\text{ind}_{a,p}(a)=1$ ，因为 $a^1 \pmod{p} = a \pmod{p} = a$;

@对于 $b = a^x \pmod{p}$

@已知 a, x, p ，计算 b 是容易的

@已知 a, b, p ，计算 x 是非常困难的

42

Diffie-Hellman密钥交换算法

39

离散对数

@对数是指数的反函数： $b = a^i \Rightarrow i = \log_a b$

@本原根 (Primitive Root)

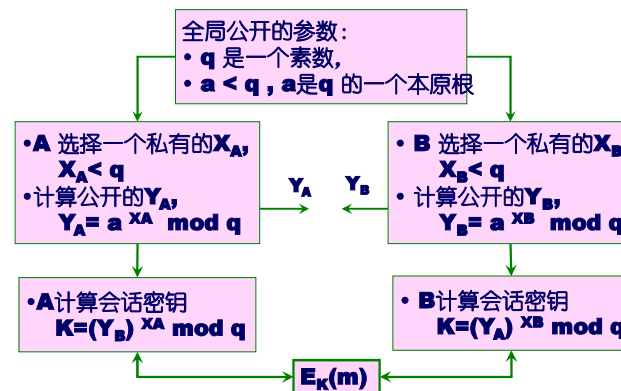
@ a 是素数 p 的一个本原根，如果

@ $a \pmod{p}, a^2 \pmod{p}, \dots, a^{p-1} \pmod{p}$ 是1到 $p-1$ 的排列，即各不相同，是整数1到 $p-1$ 的一个置换。

$p=19, a^i \pmod{p}, i=1,2,3,\dots,18$																	
a	a^2	a^3	a^4	a^5	a^6	a^7	a^8	a^9	a^{10}	a^{11}	a^{12}	a^{13}	a^{14}	a^{15}	a^{16}	a^{17}	a^{18}
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	4	8	16	13	7	14	9	18	17	15	11	3	6	12	5	10	1
3	9	8	5	15	7	2	6	18	16	10	11	14	4	12	17	13	1
4	16	7	9	17	11	6	5	1	4	16	7	9	17	11	6	5	1
5	6	11	17	9	7	16	4	1	5	6	11	17	9	7	16	4	1

41

Diffie-Hellman 密钥交换过程



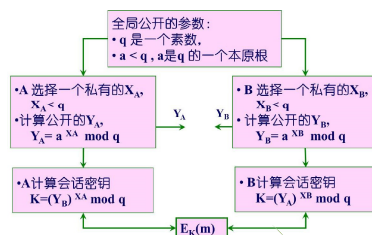
43

DH算法举例

@全局公开参数: $q=97$, $a=5$
(5是97的本原根)

@A选择私钥 $X_A=36$

@B选择私钥 $X_B=58$



④ A 计算公钥 $Y_A = 5^{36} \bmod 97 = 50$

④ B 计算公钥 $Y_B = 5^{58} \bmod 97 = 44$

④ A 与 B 交换公开密钥

@A 计算会话密钥

@ $K = Y_B^{X_A} \bmod q = 44^{36} \bmod 97 = 75$

@B 计算会话密钥

@ $K = Y_A^{X_B} \bmod q = 50^{58} \bmod 97 = 75$

44

其他公钥密码算法



@椭圆曲线密码系统

@有限域 $GF(2^n)$

@运算器容易构造

@加密速度快

@更小的密钥长度实现同等的安全性

@RSA是事实上的标准

46

其他公钥密码算法

@DSA

@1991年, NIST 提出了数字签名算法(DSA),并把它用户数字签名标准(DSS)

@DSA只能用于数字签名, 算法的安全性也是基于计算离散对数的难度

@但是DSA招致大量的反对:

④ DSA 不能用于加密或密钥分配

④ DSA是由 NIST研制的, 可能有后门

④ DSA的选择过程不公开, 提供的分析时间不充分

④ DSA比RSA慢(10—40倍)

④ 密钥长度太小(512位)

④ DSA可能侵犯其他专利

45

密码学基础

@密码学的基础知识

@对称密钥密码算法

@公开密钥密码算法

@**密钥的分配**

@情况一：传统的对称密码分配

@情况二：非对称密码中的公钥分配

@情况三：公钥密码用于传统密码体制的密钥分配

2

密码功能的设置

@对于网络而言，攻击者无处不在

@加密是所使用的最有力也最常用的方法

@需要确定的是：加密什么和在什么地方加密

@存在两种方法：

@链路加密

@端到端加密

4

密钥的分配



@无论是对称密码还是非对称密码，任何密码系统的强度都和密钥分配方法有关

@密钥分配方法就是将密钥发放给希望交换数据的双方而不让别人知道的方法

@情况一：传统的对称密码分配，要求消息交换双方共享密钥，并且此密钥不为他人所知；密钥还需要经常更换，以防攻击者知道

@情况二：非对称密码中有公钥的分配

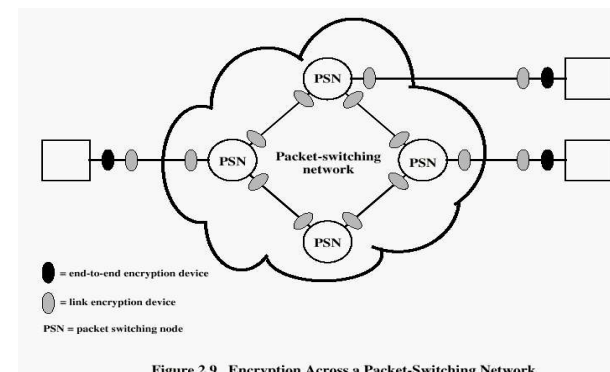
@情况三：公钥密码用于传统密码体制的密钥分配

6

密钥的分配

3

密码功能的设置



5

情况一：传统的对称密码分配

7

传统的对称密码分配

- @对参与者A和B，传统的对称密钥分配有：
 - ①方法一：密钥由A选择，并亲自交给B
 - ②方法二：第三方选择密钥后亲自交给A和B
 - ③方法三：如果A和B以前或者最近使用过某个密钥，其中一方可用它加密一个新密钥后再发送给另外一方
 - ④方法四：A和B与第三方C均有秘密渠道，则C可以将一密钥分别秘密发送给A和B
- @方法一和二需要人工传送密钥，适用于链路加密；对于端到端加密，则使用密钥分配中心

8

密钥分配中心KDC模式

- @A和B获得一次性会话密钥Ks的工作过程
 - ①STEP1：A向KDC请求一个会话密钥以保护与B的逻辑连接；消息中有A和B的标识以及一个临时交互号N1
 - ②STEP 2：KDC用Ka加密的消息做出响应；此时，A可知：
 - ①一次性会话密钥Ks，用于会话
 - ②原始请求消息，含N1，使A可以做出反映
 - ③此外，还含有两项给B的内容，用Kb加密了
 - ①一次性会话密钥Ks 和 A的标识符IDa
 - ③STEP 3：A存下会话密钥Ks备用，将KDC消息中的后两项内容发给B，B也知道会话密钥了
- @这样会话密钥就安全地发给了A和B

10

密钥的控制



- @在网络规模很大的时候，密钥的分配功能不限定在单个的KDC上面，而是使用层次式的KDC
- @层次式KDC密钥分配使得主密钥分配的代价变小了。而且，如果一个本地KDC出错，或者被攻击了，破坏只是集中在一个区域中，不会影响全局

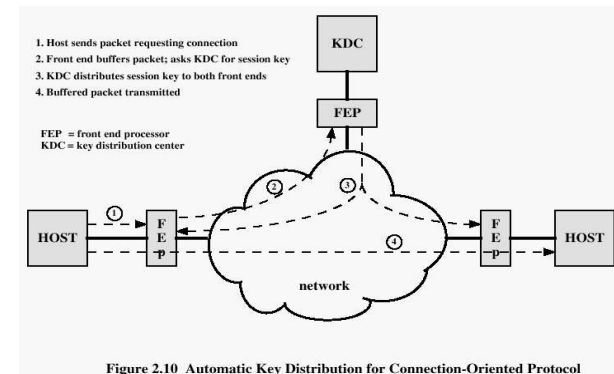
12

密钥分配中心KDC模式

- @假定每个用户与密钥分配中心KDC共享唯一的一个主密钥
 - ①设A要和B建立一个逻辑连接
 - ②A有一个除了自己只有KDC知道的主密钥Ka
 - ③B有一个除了自己只有KDC知道的主密钥Kb

9

面向连接协议的自动密钥分配



11

情况二：公钥的分配

13

公钥的分配

@公钥密码的主要作用之一就是解决密钥分配问题，公钥密码可用于下面两个方面：

- @公钥的分配
- @公钥密码用于传统密码体制的密钥分配

@常用的公钥分配方法有四种：

- @公开发布
- @公开可访问目录
- @公钥授权
- @公钥证书

14

方法二：公开可访问目录

@维护一个动态可访问的公钥目录可以获得更大程度的安全性；某个可信的实体或者组织负责这个公开目录的维护和分配

- @管理员通过对每个通讯方建立一个目录项<姓名，公钥>来维护目录，管理员定期发布或者更新该目录
- @每个通讯方通过目录管理员注册一个公钥；通讯方在任何时候都可以用新的密钥替代当前密钥
- @通讯方也可以访问电子目录；当然，它必须拥有从管理员到他的安全认证通道

@这种方法比公开公钥要安全，但是也存在缺点。一旦攻击者获得或者计算出目录管理员的私钥，带来的危险将很大

16

方法三：公钥授权

@公钥授权的方法很好，但其缺点在于只要用户之间通信，就必须向目录管理员申请对方的公钥，因此公钥管理员就会成为系统的瓶颈

18

方法一：公开发布

@任一通信方可以将他的公钥发送给另外一个通讯方或者广播给通信各方

- @例如：目前在电子邮件的认证和保密性方面广泛应用的PGP(pretty good privacy)协议，很多PGP用户就在自己发送的信息之后加入自己的公钥

@这种方法比较简便，但是它有一个大缺点，就是任何人都可以伪造这种公钥的公开发布

15

方法三：公钥授权

@通过更严格地控制公钥分配，可以使得公钥分配更加安全

@公钥授权是一种更严格的方法，它的工作过程：

- @A发送一条带有时间戳的消息给公钥管理员，请求B的当前公钥
- @管理员给A发送一条用其私钥KR加密的消息，A可以用管理员的公钥解密。这条消息中包含：B的公钥、原始请求、原始时间戳
- @A保存B公钥，并将A的表示和临时交互号N1发给B
- @与A一样，B使用同样的方法从管理员那里得到A的公钥。
- @A与B已经可以通信了，再通过核对临时交互号确认各自的身份后，安全的通信机制就建立了

17

方法四：公钥证书



@Kohnfelder最早提出不通过管理员，而用整数来交换密钥；此方法与公钥授权的安全性相同

@证书包含者公钥和其它一些信息，由证书管理员产生，并发给拥有相应私钥的通讯方

@通信一方通过传递证书将密钥信息传递给另外一方，其它通信各方可以验证该证书确实是由证书管理员发出的

@这种方法需要满足下面的要求：

- @任何通信方可以读取证书并确定证书拥有者的姓名和公钥
- @任何通信方可以验证该证书出自证书管理员，而不是伪造的
- @只有证书管理员才可以产生并更新证书
- @任何通信方可以验证证书的当前性

19

情况三： 利用公钥分配传统密码的密钥

20

简单的密钥分配

- @Merkle提出了一种及其简单的方法。例如：A要和B通信，则执行：
 - @A产生公/私钥对 $\{KU_a, KR_a\}$ ，并将含有 KU_a 和A表示的消息发给B。
 - @B产生秘密钥 K_s ，用A的公钥对 K_s 加密后传给A。
 - @A计算 $D_{KR_a}[E_{KU_a}[K_s]]$ 得到秘密钥 K_s 。
 - @这样A和B就可以利用 K_s 和对称密码进行安全通信。

@不过，这种协议容易受到主动攻击

22

具有保密性和真实性的密钥分配



- @A选择密钥 K_s ，并将 $M = E_{KU_b}[E_{KR_a}[K_s]]$ 发送给B
 - @B计算 $D_{KU_a}[D_{KR_b}[M]]$ 得到密钥 K_s
 - @A和B可以利用对称密码进行安全通信了
- @混合方法
- @利用公钥密码进行密钥分配
 - @这种方法也需要密钥分配中心KDC，KDC与每个用户共享一个主密钥，通过该主密钥加密实现会话密钥的分配

24

利用公钥分配传统密码的密钥

@由于公钥密码速度较慢，几乎没有用户愿意在通信中完全使用公钥密码，因此公钥密码更适合作为传统密码中实现密钥分配的一种手段。

- @下面介绍三种密钥分配方法
 - @简单的密钥分配方法
 - @具有保密性和真实性的密钥分配方法
 - @混合方法

21

具有保密性和真实性的密钥分配

- @下面这种方法即可以抗击主动攻击也可以抗击被动攻击
 - @假设A和B已经交换了公钥
 - @A用B的公钥对含有A标识和临时交互号 N_1 的消息加密，并发给B
 - @B发送一条用A的公钥加密的消息，包括A的 N_1 和B产生的临时交互号 N_2
 - @因为只有B能够解密A发来的消息，所有本条消息中的 N_1 可以使A确信其通信伙伴是B
 - @A用B的公钥对 N_2 加密，并返回给B，这样可以使B确信其通信伙伴是A

23

课程主要内容

- @一、计算机网络安全概论
- @二、密码学基础
 - @密码学的基础知识
 - @对称密钥密码算法
 - @公开密钥密码算法
 - @密钥的分配
- @三、认证技术
 - @消息认证技术
 - @身份认证及其实例
- @四、网络安全协议
- @五、各类专用技术

认证要求

网络环境中的攻击

- @内容修改：对消息内容的修改，包括插入、删除、转换和修改。
- @顺序修改：对通信双方消息顺序的修改，包括插入、删除和重排。
- @计时修改：对消息的延时和重放。
- @发送方否认：发送方否认发送过某条消息。
- @接收方否认：接收方否认接收过某条消息。
- @对付泄密、传输分析属于消息**保密性**范畴。
- @对付伪装、内容修改、顺序修改、计时修改属于**消息认证**范畴。
- @对付发送方否认可以使用**数字签名**。
- @对付接收方否认需要使用**数字签名**和为抗击此类攻击而设计的协议。

消息认证技术 Message Authentication

- @认证要求
- @认证函数
 - @消息加密
 - @消息认证码MAC
 - @Hash函数
- @Hash算法
 - @安全hash函数的一般结构
 - @主要Hash算法概述
 - @MD5算法
 - @SHA算法
 - @RIPEMD-160算法
 - @各类算法比较
- @数字签名

网络环境中的攻击

- @泄密：将消息透露给没有合法密钥的接收方。
- @传输分析
 - @分析通信双方的通信模式。
 - @在面向连接的应用中，确定连接的频率和持续时间。
 - @在面向连接或无连接的环境中，确定双方的消息数量和长度。
- @伪装
 - @欺诈源向网络中插入一条消息。
 - @攻击者产生一条消息并声称这条消息来自某个合法实体。
 - @非消息接收方发送关于收到或者未收到消息的欺诈应答。

消息认证的概念



- @消息认证就是验证所收到的消息确实是来自真正的发送方且未被修改的消息。
 - @消息认证也可以验证消息的顺序和及时性。
 - @数字签名是一种认证技术，可以用来抗击发送方否认。
 - @各种认证协议属于认证技术，被应用到各个方面。
- @任何消息认证在功能上面可以看作两层：
 - @下面一层中一定有某种产生认证符的函数，**认证符**是一个用来认证消息的值。
 - @上面协议中将该函数作为原语使接收方可以验证消息的真实性。

认证函数

- @消息加密
- @消息认证码
- @Hash函数

认证函数 之 消息加密

对称加密

- @发送方A用A和B共享的密钥K对发送到接收方B的消息M进行加密，如果没有其它地方知道该密钥，就可以提供保密性。
- @同时，B可以在接收到消息后，用密钥K解密，确认该消息是由A发出的。但这不是绝对的。B需要确认密文是否真正来自A。
- @解决对称加密中消息认证的方法是要求明文具有某种易于识别的结构，而且，不通过加密函数是不能重复这种结构的。

认证函数



- @产生认证符的函数称为认证函数，认证函数可以分成三类：
 - @消息加密：整个消息的密文作为认证符。
 - @消息认证码MAC：MAC是消息和密钥的公开函数，它产生定长的值，该值作为认证符。
 - @Hash函数：它是将任意长的消息映射为定长的hash值的公开函数，以该hash值作为认证符。

消息加密

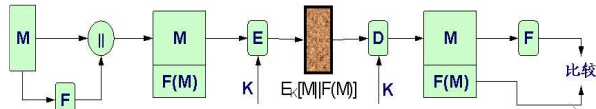
- @消息加密本身提供了一种认证手段。
- @传统密码和公钥密码体制中，对消息加密的分析是不同的。

对称加密的例子

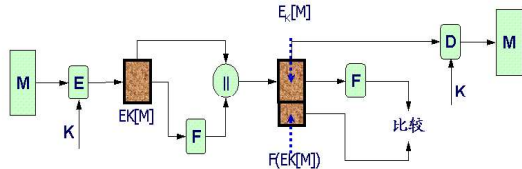
- @加密前对每个消息附加一个错误检测码，也称之为帧校验序列FCS或者校验和。
 - @A准备发送明文消息M，那么A将M作为函数F的输入，产生FCS，将FCS附加在M后并对M和FCS一起加密。
 - @在接收端，B解密其收到的信息，并将其看作是消息和附加FCS。B用相同的函数F重新计算FCS，若所得FCS和收到FCS相等，B认为消息是真实的。
 - @FCS和加密函数的顺序是很重要的。

对称加密的例子

@先计算FCS，再加密。可以提供认证。



@先加密，再计算FCS。攻击者可以构造具有正确错误控制码的消息，造成混淆。



公钥加密

@公钥加密只提供保密性，不能提供认证。

@如果既要提供保密性，又要提供认证，发送方A可以先用其私钥加密(数字签名)，然后用B的公钥加密。

@这种方法的缺点是执行了四次附加的公钥算法运算。

认证函数之消息认证码MAC

对称加密的例子

@例如：TCP协议的传输。

- ③可以对除了IP报头之外的所有数据进行加密。
- ③如果攻击者利用一条消息代替加密后的TCP段，解密后也不能恢复出原来的IP报头。
- ③这种方法中，头不仅包含了校验和，而且还有序列号，用于防止延时、删除段和改变段的顺序。

32bit	
源端口 Source Port	目的端口 Destination Port
序列号 Sequence Number (32bit)	
确认号 Acknowledgement Number (32bit)	
报头长度 (4bit)	窗口大小 Window Size (16bit)
保留 (6bit)	紧急指针 Urgent Pointer
ACK	PSH
URG	SYN
FIN	
校验和 Checksum	
可选项 (0~32bit)	
数据 Data(optional)	

各种加密方法的特点



@对称加密

- ③提供保密性
 - ③只有A和B共享密钥K
- ③提供认证
 - ③只能发自A
 - ③传输中未被改变
 - ③需要某种数据形式/冗余
- ③不能提供数字签名
 - ③接收方可以伪造信息
 - ③发送方可以否认信息

@公钥加密

- ③提供保密性
 - ③只有B拥有用于解密的密钥 K_{Bb} ，但是任何一方都可用 K_{Ub} 对消息加密并假称是A
- ③提供认证和签名
 - ③只有A拥有用于加密的密钥 K_{Ra}
 - ③传输中未被改变
 - ③需要某种数据组织形式/冗余
 - ③任何一方可用 K_{Ua} 来验证签名
- ③提供保密性、认证+签名
 - ③提供保密性（因为 K_{Ub} ）
 - ③提供认证和签名（因为 K_{Ra} ）

消息认证码

@消息认证码MAC也是一种认证技术，它利用密码生成一个固定长度的短数据块，并将该数据块附加在消息之后。

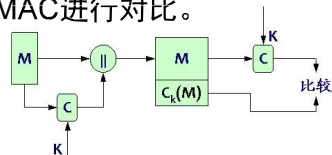
@MAC是消息和密钥的函数： $MAC=C_k(M)$

- ③M：输入消息；
- ③C：MAC函数
- ③K：收发双方的共享密钥
- ③MAC：消息认证码

@MAC函数与加密类似，区别就是MAC算法不要求可逆性，而加密算法必须可逆。

MAC认证的过程

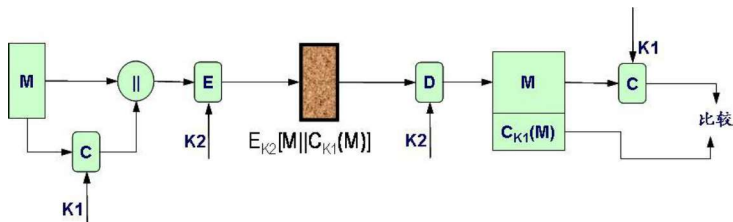
- @A和B共享密码K。
- @A向B发送消息时，A计算MAC，将其附加在消息后面，一起发送给接收方。
- @接收方对收到的消息用相同的密钥K进行计算得到MAC，并与收到的MAC进行对比。



- @因收发双方共享密钥，MAC不能提供数字签名。

认证 + 保密

- @在实际中，一般将认证和保密性相结合使用。
 - @与明文有关的认证：先将消息作为输入，计算MAC，附加在消息之后，然后对整个信息块加密。



使用消息认证码的情况



- @将同一消息广播给很多接收者的情况。
 - @一个接收者负责验证真实性，告知其它人。
- @在信息交换中，接收者希望可以随机地对消息进行认证。
- @对明文形式的计算机程序进行认证是很有意义的服务。
 - @运行一个计算机程序，不需要每次都进行加密解密。只在需要保护程序完成性的时候才检验消息认证码MAC。
- @一些应用只关心消息的认证，而不关心消息的保密性。
 - @例如SNMPv3
- @将认证和保密性分开，可以使层次结构更灵活。
 - @例如可以在应用层提供认证，在传输层提供保密性。

MAC认证的过程

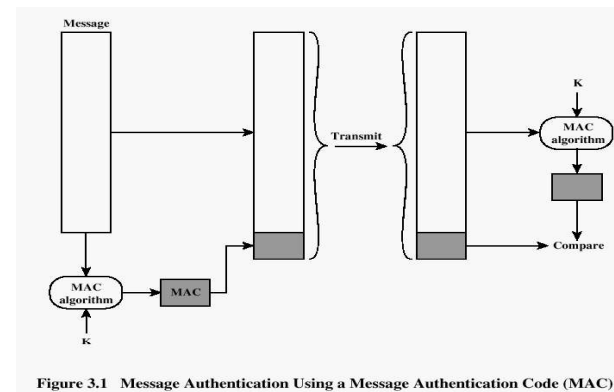
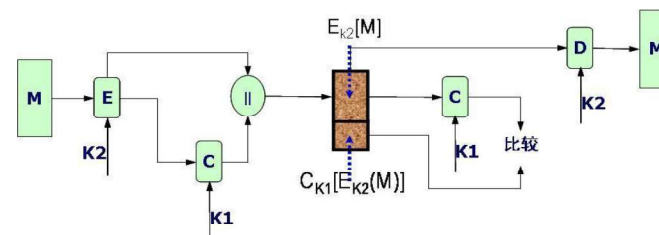


Figure 3.1 Message Authentication Using a Message Authentication Code (MAC)

认证 + 保密

- @与密文有关的认证：先将消息加密，然后将密文作为输入，计算MAC，并将MAC附加在上述密文之后形成待发送的信息块。



认证函数之 Hash函数

Hash函数

@单向的hash函数是消息认证码的一种变形。与消息认证码一样，hash函数的输入是大小可变的消息M，输出是固定大小的hash码H(M)。

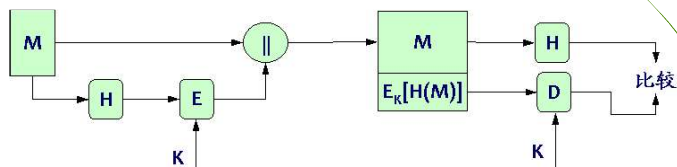
@与MAC不同的是，hash码并不使用密钥，它仅仅做为输入消息的函数，又是hash码也称为消息摘要(Message Digest, MD)。

@Hash码是所有消息位的函数，它具有错误检测能力，即改变消息的任何一位或者多位，都会导致hash码的改变。

将hash码用于消息认证2

@用对称密码仅对hash加密。对不要求保密性的应用，这种方法会减少处理代价。

@此种方法提供了认证。

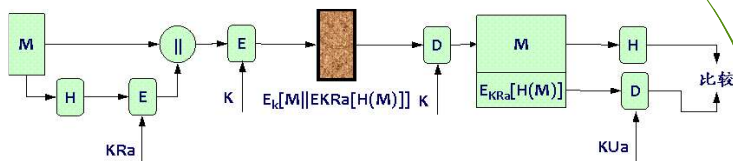


@Hash函数和加密函数合成就是MAC。

将hash码用于消息认证4

@即希望保证保密性又希望有数字签名的，先用发送方的私钥对hash码进行加密，再用对称密码中的密钥对消息和上述加密结果进行加密。

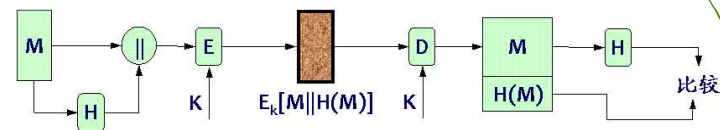
@这种技术比较常用。提供认证、数字签名和保密性。



将hash码用于消息认证1

@用对称密码对消息及附在其后的hash码进行加密。由于只有A和B共享密钥，所以消息一定是来自A且未被修改过的。

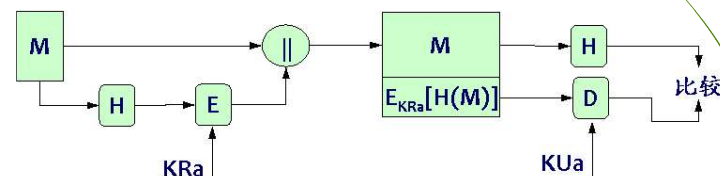
@Hash码提供了认证所需要的结构或者冗余，并且是对整个消息和hash码加密，所以也提供了保密性。



将hash码用于消息认证3

@用公钥密码和发送方的私钥对hash码加密。

@这种方法可以提供认证，由于只有发送方可以产生加密后的hash码，所以这种方法也提供了数字签名。

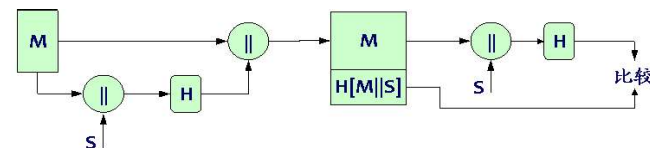


将hash码用于消息认证5

@假定通信双方共享公共的秘密值S，A将M和S联接后再计算hash值，并将其附加在M后面。由于B也知道S，所以B可以计算hash值，并验证其正确性。

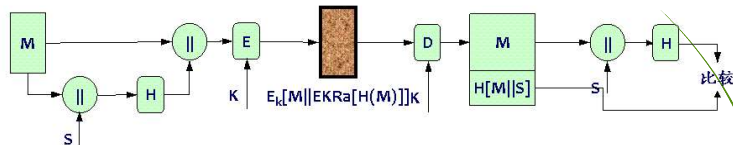
@由于秘密值S不在网络中传送，所以攻击者无法修改所截获的消息，也不能伪造消息。

@提供认证。



将hash码用于消息认证6

- @对整个消息和hash码加密。
- @提供认证和保密性。



对Hash函数的要求



- @Hash：哈希函数，杂凑函数，散列函数 $h = H(m)$
- @H 具有如下特性：
 - @可以操作任何大小的报文m
 - @给定任意长度的m，产生的h的长度固定
 - @给定m 计算 $h = H(m)$ 是容易的
 - @给定h, 寻找m, 使得 $H(m) = h$ 是困难的【单向性】
 - @给定m, 要找到 $m', m' \neq m$ 且 $H(m) = H(m')$ 是计算上不可行的【抗弱碰撞性】
 - @寻找任何 (x, y) , $x \neq y$, 使得 $H(x) = H(y)$ 是计算上不可行的【抗强碰撞性】

安全hash函数的一般结构

- @典型的安全hash函数的一般结构是由Merkel提出的，又被称为迭代hash函数。
- @MD5、SHA-1和RIPEMD-160等目前广泛使用的hash函数都是采用这种结构的。
- @Hash函数将输入消息分为L个固定长度的分组，每个分组长度为b位，最后一个分组不足b位时，需要填充成b位。
- @输入中包含长度，因此增加了攻击的难度。因为，攻击者必须找出
 - @具有相同hash值且长度相等的两条消息
 - @或者，找出两条长度不等但加入消息长度后hash值相同的消息

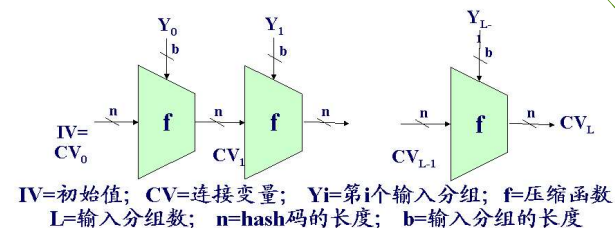
不使用加密函数

- @如果不要求保证保密性，就可以不使用加密函数，从而减少计算代价。
- @基于以下原因可以考虑不使用加密函数：
 - @加密软件速度慢。
 - @加密硬件成本不容忽视。
 - @加密硬件的优化是针对大数据块的。小数据块则在初始化和调用上面花费了大量时间。
 - @加密算法可能收到专利保护。
 - @加密算法受到美国出口限制。

Hash算法之 安全hash函数的一般结构

安全hash函数的一般结构

- @Hash函数公式
 - @ $CV_0 = IV =$ 初始n位值
 - @ $CV_i = f(CV_{i-1}, Y_i)$ $1 \leq i \leq L$
 - @ $H(M) = CV_L$ (其中输入M由 $Y_0, Y_1, \dots, Y_{L-1}$ 组成)



安全hash函数的一般结构

- @Hash函数中重复使用了压缩函数f。
 - ②它的输入是前一步中得出的n位结果（即连接变量）和一个b位分组，输出位一个n位分组。
 - ②连接变量的初始值在算法开始的时候指定，其终值即为hash值。通常 $b > n$ ，因此称为压缩函数。
- @如果压缩函数具有抗碰撞能力，那么迭代hash函数也具有抗碰撞能力。
- @由此可见，设计安全hash函数可以归纳为设计具有抗碰撞能力的压缩函数问题，并且该压缩函数的输入是定长的。

Hash算法之 主要Hash算法概述

Hash算法— MD族

- @MD4
 - ②1990年开发出MD4算法。
 - ②有人很快发现了MD4版本中第一、三步的漏洞，并演示了如何利用一部普通的个人电脑在几分钟内找到MD4完整版本中的碰撞。
- @MD5
 - ②1991年，Rivest开发出更为成熟的MD5算法。
 - ②曾有人发现MD5算法中的伪碰撞，后来被王小云攻破了。
- @RIPEMD-128/160/320
 - ②RIPEMD由欧洲财团开发和设计的MD算法。

安全hash函数的一般结构



- @因为hash函数中， $b > n$ ，将分组大小为b位的消息映射为长度为n的hash码，所以，任何hash函数都存在碰撞，因此要求找出在计算上面不可行的碰撞。

Hash算法— MD族

- @MD = Message Digest（消息摘要）
- @90年代初，MD族哈希函数是在由Ron · Rivest设计成功的。MD2、MD4和MD5都产生一个128位的消息摘要。
- @MD2
 - ②1989年开发出MD2算法。
 - ②算法中，首先进行数据补位，使字节长度是16的倍数；然后以一个16位的检验和追加到末尾；最后计算出散列值。
 - ②后来Rogier和Chauvaud发现：如果忽略了检验和，将产生MD2碰撞。

Hash算法— SHA族

- @SHA = Secure Hash Algorithm
- @SHA系列算法是NIST根据Rivest设计的MD4和MD5开发的算法；美国国家安全局发布SHA作为政府标准。
 - ②SHA-0：正式地称作SHA系列，发行后不久即被指出存在弱点
 - ②SHA-1：1994年发布的，与MD4和MD5散列算法非常相似，被认为是MD4和MD5的后继者。
 - ②SHA-2：实际上分为SHA-224、SHA-256、SHA-384和SHA-512算法。
 - ②SHA-3：方案正在征集中。

Hash算法— 其它

@HAVAL：加密哈希算法

- ①1992年，由Yuliang Zheng, Josef Pieprzyk, and Jennifer Seberry三人设计。与MD5结构不同，更象现代加密散列函数。
- ②HAVAL可以产生不同长度的哈希值128位，160位，192位，224位和256位；还允许用户指定轮数（3、4或5）。

@Gost

- ①Gost是一套俄国标准。

Hash算法之 MD5 --- Message Digest

MD5 算法的设计目标

@安全性

- ①找到两个摘要相同的消息在计算上不可行。

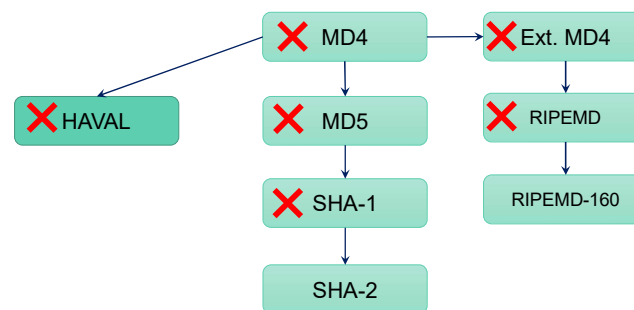
@速度

- ①算法应该有利于快速的软件实现。特别是，算法的快速实现是针对32位机的，因此算法基于的是字长为32位的基本操作。

@简单和简洁性

- ①算法应易于描述且易于编程，不需要使用大程序或者置换表。

主要Hash算法的发展历程



MD5 算法

@MD5消息摘要算法(RFC 1321)是由 MIT Laboratory of Computer Science 的Ron Rivest 提出的。

@MD5的输入是任意长度的消息，对输入按照512位的分组为单位进行处理，算法的输出是128位的消息摘要。

- ①输入:任意长的消息
- ②分组:512比特
- ③输出:128比特

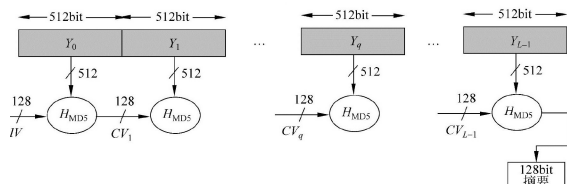
MD5 算法的设计目标

@倾向于使用低端结构

- ①某些处理器(如Intel 80xxx和Pentium系列)将字的最低有效字节存于低地址字节位置（低端），而其它的处理器(如Sun Sparcstation)将字的最高有效位存于低地址字节位置（高端）。
- ②将消息作为32位的字序列进行处理的时候，这两种处理方法有着明显的区别，因为其中一种结构需要将每个要处理的字节反向。
- ③Rivest注意到使用高端结构的处理器一般比较快，因而能够承受这种处理代价，所以选择使用低端结构将消息表示为32位的字序列。

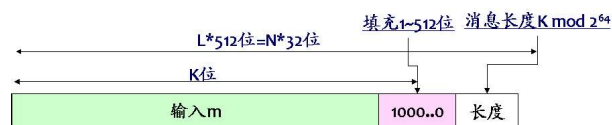
MD5算法步骤

- @Step1: 增加填充位。
- @Step2: 填充长度。
- @Step3: 初始化MD 缓存。
- @Step4: 以512位的分组(16个字)位单位处理消息。
- @Step5: 输出。



Step1 : 增加填充位

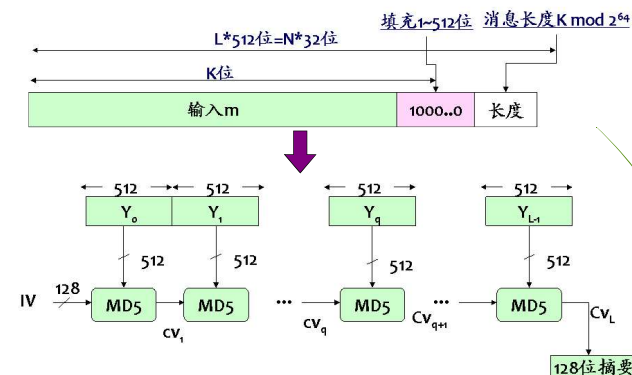
- @填充比特: 第1位为1, 其后各位皆为0。
- @填充比特后, 使之与448模512同余;
即填充后的消息比512的整数倍少64位。



Step3 : 初始化MD 缓存

- @Hash函数的中间结果和最终结果都保存于128位的缓冲区中, 缓冲区用4个32位的寄存器(A,B,C,D)表示。
- @A,B,C,D寄存器以低端格式存储, 其初始值为:
 - @A= 01234567
 - @B= 89ABCDEF
 - @C=FEDCBA 98
 - @D=76543210

MD5 算法过程



Step2 : 填充长度

- @用64位表示填充前的报文长度, 附加填充比特的后面。如果长度大于 2^{64} , 则对 2^{64} 取模。
- @以little-endian方式表示被填充前的消息长度。
 - @Little-endian方式指按数据的最低有效字节优先的顺序存储数据, 即将低有效字节存于低地址字节。相反的存储方式为big-endian方式。
 - @增加消息长度的目的是为了加强算法安全强度。
- @完成比特填充和长度填充后, 消息的总长度是512的整数倍。

Step4 : 计算Hash值

- @以512位的分组(16个字)为单位处理消息。
 - @由四轮运算组成的压缩函数是算法的核心。压缩函数标记为HMD5。
 - @使用一个随机矩阵

$$T[i] = 232 \cdot \text{abs}(\sin(i)) \quad i=1,2,\dots,64$$

Step5 : 输出

@Step5: 输出。

②所有的L个512位的分组处理完之后，第L个分组的输出即是128位的消息摘要。

@MD5算法的公式表示

②CV0= IV

②CVq+1= SUM32 (CVq, RFI[Yq ,RFH[Yq , RFG[Yq ,RFF[Yq ,CVq]]])

②MD=CVL-1

MD5 的压缩函数

②压缩函数由四轮运算构成，四轮运算结构相同，但各轮使用不同的基本逻辑函数，分别称之为F/G/H/I。

②每轮的输入是当前要处理的512位的分组(Yq)和128位缓冲区ABCD的内容。

②表T有64个元素，每轮使用表T[1...64]中的16个元素，并更新缓冲区。

②表T是通过正弦函数构造的，T的第i个元素记为T[i]。T的每个元素都可以用32位表示，堆积化的32位输入数据，消除了输入数据的规律性。

②T[i]= 232 abs (sin (i)) i=1,2,..64

②第四轮的输出与第一轮CVq的输入相加得到CVq+1。

②加法是指缓冲区中的4个字与CVq中对应的4个字分别模2³²相加。

MD5 每轮处理512位分组的过程

②MD5中每轮对缓冲区ABCD进行16步迭代，每步迭代为：

$b \leftarrow a + ((a + g(b,c,d) + X[k] + T[i]) \lll s)$

②a,b,c,d：缓冲区的四个字，它按照一定的次序随迭代步变化

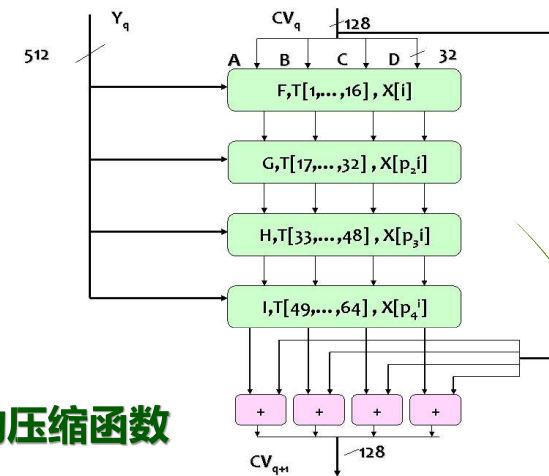
②g：基本逻辑函数F/G/H/I之一

② $\lll s$ ：32位的变量循环左移s位

②X[k] = M[q*16+k]：消息第q个512位分组的第k个32位字

②T[i]：矩阵T中的第i个32位字

②+：模232加法



MD5 的压缩函数

从正弦函数构造的表T

T[1]=D76AA478	T[17]=E76A2562	T[33]=FFFA3492	T[49]=4BDF7221
T[2]=E8C7B756	T[18]=756FDA1	T[34]=8871F681	T[50]=
T[3]=	T[19]=	T[35]=	T[51]=
T[4]=	T[20]=	T[36]=	T[52]=
T[5]=	T[21]=	T[37]=	T[53]=
... ..			
T[16]=	T[32]=	T[48]=	T[64]=

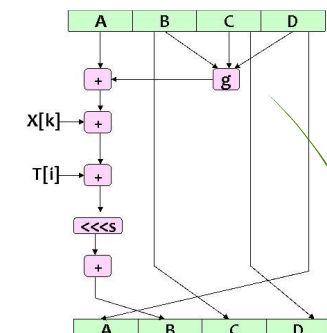
MD5 每轮处理512位分组的过程

② $b \leftarrow a + ((a + g(b,c,d) + X[k] + T[i]) \lll s)$

②X[k]：消息第q个512位分组的第k个32位字

②T[i]：矩阵T中的第i个32位字

循环	原始函数	g(b,c,d)
1	F	(b^c) v (~b^d)
2	G	(b^d) v (c^~d)
3	H	b xor c xor d
4	I	c xor (b v ~d)



MD5的强度

@MD5算法中，hash函数的每一位都是输入的每一位的函数。基本逻辑函数的复杂迭代使得输出对输入的依赖性非常小，即随机选择的两条消息即是具有相同的规律性，也不可能产生相同的hash码。

@Ron Rivest猜测，MD5可能是128位的hash码中最强的算法，找到两条hash码相同的消息的代价是 2^{64} 数量级，找到具有给定摘要的消息所需代价为 2^{128} 数量级。

MD4算法



@MD4和MD5算法的主要区别在于：

- @MD4使用三轮运算，每轮16步迭代；MD5使用四轮运算，每轮16步迭代。
- @MD4第一轮运算没有使用加法常量，第二轮运算中每步迭代使用的加法常量相同，第三轮运算中每步迭代使用的加法常量也相同，但不同于第二轮的加法常量；MD5的64步迭代中，每步使用不同的加法常量 $T[i]$ 。
- @MD5使用四个基本逻辑函数，每轮运算使用一个基本逻辑函数；MD4使用三个基本逻辑函数，每轮运算使用一个基本逻辑函数。
- @MD5中的每步迭代结果都与前一步的结果相加，MD4中没有如此。

Hash算法之 SHA — Secure Hash Algorithm

MD4算法

@MD4和MD5算法都是Ron Rivest提出的，MD4出现在MD5之前。

@MD4最初作为RFC发表于1990年10月，其修订版和MD5同时发表于1992年4月的RFC 1320中。

石破惊天 MD5堡垒轰然倒塌

安全hash算法 — SHA

@安全hash算法是NIST设计，于1993作为联邦信息处理标准FIPS180发布的，修订版于1995年发布FIPS 180-1，也称之为SHA-1。RFC 3174也给出了SHA-1。

@SHA算法建立在MD4之上，基本框架与MD4类似。

@SHA-1算法输入是长度小于 2^{64} 位的消息，输出是160位的消息摘要，输入消息以512位的分组为单位进行处理。

@SHA-1算法也将消息按照512位分组，但hash值和连接变量长为160位。

SHA-1算法步骤

@Step1: 增加填充位。

@填充消息使之与448模512同余。即填充后的消息比512的整数倍少64位。

@Step2: 填充长度。

@用64位表示填充前的报文长度，附加在填充后的结果后面。

@Step3: 初始化MD 缓存。

@Hash函数的中间结果和最终结果都保存于160位的缓冲区中，缓冲区用5个32位的寄存器(A,B,C,D,E)表示，并将这些寄存器初始化。

SHA-1压缩函数

@处理一个512位的分组要执行80步。每步的处理过程是一样的。

@A,B,C,D,E: 缓冲区的5个字

@t: 步骤编号, $0 \leq t \leq 79$

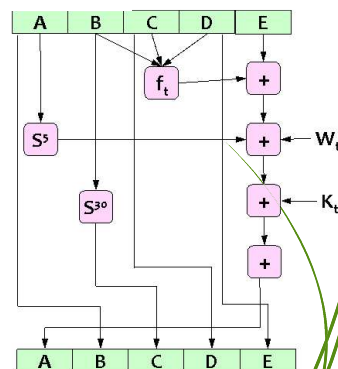
@ $f_t(B,C,D)$: 第t步使用的基本逻辑函数

@Sk: 32位的变量循环左移k位

@Wt: 从当前512位输入分组导出的32位字

@Kt: 加法常量。共使用了四个不同的加法常量。

@+: 模 2^{32} 加法



Hash算法之 RIPEMD-160算法

SHA-1算法步骤

@Step4: 以512位的分组(16个字)位单位处理消息。

@由十轮运算组成的压缩函数是算法的核心，十轮运算分成两组，每轮执行16迭代20步。

@Step5: 输出。

@所有的L个512位的分组处理完之后，第L个分组的输出即是160位的消息摘要。

@SHA-1算法的公式表示

@CV0= IV

@CV_q+1= SUM32 (CV_q, ABCDE_q)

@MD=CVL

SHA系列算法碰撞攻击复杂度



Algorithm	Output size (bits)	Internal state size (bits)	Block size (bits)	Max message size (bits)	Word size (bits)	Rounds	Operations	Collision
SHA-0	160	160	512	$2^{64} - 1$	32	80	+, and, or, xor, rotl	Yes
SHA-1	160	160	512	$2^{64} - 1$	32	80	+, and, or, xor, rotl	2^{63} attack
SHA-256/224	256/224	256	512	$2^{64} - 1$	32	64	+, and, or, xor, shr, rotr	None
SHA-512/384	512/384	512	1024	$2^{128} - 1$	64	80	+, and, or, xor, shr, rotr	None

RIPEMD-160算法

@RIPEMD-160消息摘要算法是欧洲RIPE计划下一个研究组设计的

@这些人曾对MD4和MD5进行过一些成功的攻击。

@后来，他们和H.Dobbertin合作，一起完成了对两轮RIPEMD算法改进工作，最终形成了RIPEMD-160消息摘要算法。

@RIPEMD-160算法的输入是任意长的消息，输出是160位的消息摘要。输入以512位的分组单位进行处理，hash值和链接变量是160位。

RIPEMD-160算法步骤

@Step1: 增加填充位。

②填充消息使之与448模512同余。即填充后的消息比512的整数倍少64位。

@Step2: 填充长度。

②用64位表示填充前的报文长度, 附加在填充后的结果后面。

@Step3: 初始化MD 缓存。

②Hash函数的中间结果和最终结果都保存于160位的缓冲区中。

②缓冲区用5个32位的寄存器(A,B,C,D,E)表示, 并将这些寄存器初始化。

Hash算法之 MD5/SHA-1/RIPEMD-160算法比较

MD5/SHA-1/RIPEMD-160比较

@抗穷举攻击的能力

②三个算法都不易受到弱碰撞性的攻击。

②MD5由于消息摘要短, 易于收到强碰撞性的攻击。

②SHA-1和RIPEMD-160在不久的将来, 对强碰撞性的攻击还是安全的。

@抗密码分析的能力

②对MD5的密码分析取得了很大进展。

②对SHA-1的密码分析比MD5要困难, 对RIPEMD-160的密码分析比SHA-1又要困难。

RIPEMD-160算法步骤



@Step4: 以512位的分组(16个字)位单位处理消息。

②由十轮运算组成的压缩函数是算法的核心, 十轮运算分成两组, 每组五轮, 每轮执行16步迭代。

@Step5: 输出。

②所有的L个512位的分组处理完之后, 第L个分组的输出即是160位的消息摘要。

MD5/SHA-1/RIPEMD-160比较

	MD5	SHA-1	RIPEMD-160
摘要长度	128 bits	160 bits	160 bits
基本处理单元	512 bits	512 bits	512 bits
步数	64(4 轮, 每轮 16 步)	80(4 轮, 每轮 20 步)	160(5 轮, 每轮 16 步)
最大消息长度	∞	$2^{64}-1$ 位	$2^{64}-1$ 位
基本逻辑函数	4	4	5
使用的加法常量	64	4	9
低端位/高端位结构	低位在前	高位在前	低位在前

MD5/SHA-1/RIPEMD-160比较



@速度

②三个算法都基于模 2^{32} 加法和简单的位逻辑运算, 所以它们在32位机上执行速度较快。

②由于MD5相对简单, 迭代次数少, 所以运行速度最快。

@低端位/高端位结构

②MD5和RIPEMD-160采用低位在前结构, SHA-1采用高位在前结构。

DSS数字签名算法

数字签名

- @在收发双方不能完全信任的情况下，就需要其它的方法来解决，数字签名就是一个最好的解决方法。
- @数字签名是公钥密码学发展过程中最重要的概念之一，它可以提供其它方法难以实现的安全性。
- @数字签名相当于手写签名，须具备以下特征：
 - ①它必须能够验证签名者、签名日期和时间。
 - ②它必须能认证被签的消息内容。
 - ③签名应能由第三方仲裁，以解决争执。
- @数字签名可以分成两类：直接数字签名和仲裁数字签名。

仲裁数字签名

- @从发送方X到接收方Y的每条已签名的消息都先发给仲裁者A，A对消息及其签名进行检查以验证消息源及其内容，然后给消息加上日期，并发给Y，同时指明该消息已通过仲裁者的检验。
- @A的加入解决了直接数字签名的问题。

数字签名

- @消息认证可以保证通信双方不受第三方的攻击，但是它不能处理通信双方自身发生的攻击。
- @例如A使用消息认证码MAC给B发送了一条消息，可以出现下面两种情形：
 - ①B可以伪造一条消息，并声称该消息发自A。B只需产生一条消息，用A和B共享的密钥产生认证码，并将其附在消息之后，即可。
 - ②A可以否认曾经发送过某条消息。因为B可以伪造消息，所以也无法证明A确实发送过某消息。

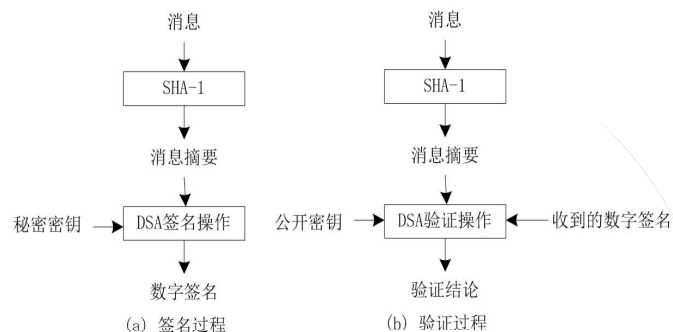
直接数字签名

- @直接数字签名只涉及通信双方。
- @假定接收方已知发送方的公钥，则发送方可以通过用自己的私钥对整个消息或者消息的hash码加密来产生数字签名。
- @之后，再用接收方的公钥(公钥密码)和共享的密钥(对称密码)对整个消息和签名进行解密，则可以获得保密性。
- @直接数字签名的弱点在于方法的有效性依赖于发送方私钥的安全性。

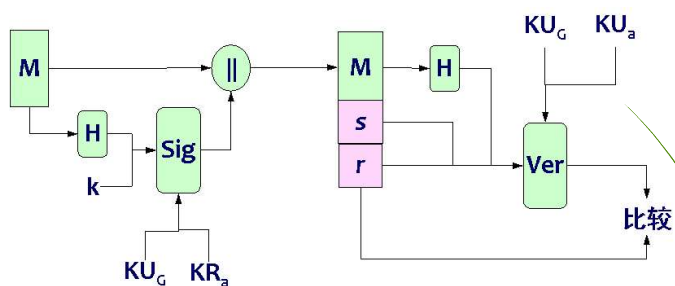
数字签名标准DSS

- @1991年8月30日，美国国家标准与技术学会NIST提出了一个联邦数字签名标准，称之为数字签名标准DSS。
 - ①1993年，根据公众对其安全性的反馈做了修改
 - ②1994年12月1日，被正式采用为美国联邦信息处理标准
 - ③1996年，又进一步修改，2000年发布了DSS的扩展版FIP186-2
- @DSS使用SHA-1算法，给出了一种新的数字签名方法，即数字签名算法DSA。
 - ①NIST提出：“此标准适用于联邦政府的所有部门，以保护未加保密的信息——它适用于E-mail、电子金融信息传输、电子数据交换、软件发布、数据存储及其他需要数据完整性和原始真实性的应用”。
- @DSS是一种公钥方法，与RSA不同，DSS不能用于加密或者密钥分配，只提供数字签名功能。

DSS的签名与验证



数字签名标准DSS



签名过程与验证过程

签名过程

用户随机选取k, 计算:

$$@r = (g^k \bmod p) \bmod q$$

$$@s = [k^{-1}(H(M) + xr)] \bmod q$$

(r, s)即为消息M的数字签名

验证过程

接收者收到M,r,s后, 首先验证 $0 < r < q$, $0 < s < q$, 如通过则计算:

$$@w = (s)^{-1} \bmod q$$

$$@u_1 = [Mw] \bmod q$$

$$@u_2 = [rw] \bmod q$$

$$@v = [(g^{u_1} y^{u_2}) \bmod p] \bmod q$$

如果 $v=r$, 则确认签名正确

数字签名标准DSS

@DSS使用hash函数, 它产生的hash码和随机数k作为数字签名函数Sig的输入, 签名函数依赖于发送方的私钥KR_a和一组通信伙伴共同拥有的参数构成的全局公钥KUG。签名函数保证只有拥有私钥的发送方才能产生有效签名。

@签名由两部分构成: s和r。

@接收方对接收到的消息产生hash码, 这个hash码和签名一起作为验证函数Ver的输入, 验证函数依赖于全局公钥KUG和发送方公钥KU_a。若验证函数的输出等于签名中的r, 则签名有效。

参数说明

@全局公钥 (p,q,g)

@p: 为L位长的素数。其中, L为512~1024之间且是64倍数的数。

@q: 是160位长的素数, 且为p-1的因子。

@g: $g = h^{(p-1)/q} \bmod p$ 。

其中, h是满足 $1 < h < p-1$ 且 $h^{(p-1)/q} \bmod p$ 大于1的整数。

@用户私钥x: x为在 $0 < x < q-1$ 内的随机数

@用户公钥y: $y = g^x \bmod p$

@用户每个消息用的秘密随机数k, $0 < k < q$

参数p、q、g是公开的; x为私钥, y为公钥;
对于每一次签名都应该产生一次k; x和k用于数字签名, 必须保密;

课程主要内容

- @一、计算机网络安全概论
- @二、密码学基础
 - @密码学的基础知识
 - @对称密钥密码算法
 - @公开密钥密码算法
 - @密钥的分配
- @三、认证技术
 - @消息认证技术
 - @身份认证及其实例
- @四、网络安全协议
- @五、各类专用技术

2

电子身份认证简介

4

互联网世界中的电子身份认证

- @（电子）身份认证是在计算机及网络中确认操作者身份的解决方法。
- @互联网世界中一切信息（包括各类对象的身份信息）都是用**一组特定的数据**来表示的，计算机只能识别用户的数字身份，因此身份认证的过程也是针对**这组特定的数据**进行的。
- @电子身份的认证方法与现实世界的身份认证方法是类似的（根据what you know、what you have、who you are），其中，需要大量使用**消息认证技术**（Message Authentication）。

6

身份认证技术主要内容

- @电子身份认证简介
 - @认证主要方法（what you know、what you have、who are you）
 - @认证技术（静态口令、动态口令、数字证书等）
- @网站身份认证技术
 - @HTTP的Basic认证
 - @基于表单Form的身份认证
 - @其它增强认证
- @其它身份认证应用
 - @操作系统认证
 - @网络接入认证

3

什么是身份认证？

- @身份认证是指在主客体交互行为过程中确认行为参与者（主体或客体，通常称为用户）身份的解决方法。
- @在现实生活中，身份认证无处不在：
 - @印章、居民身份证、银行卡及密码、飞机票、等等……
- @在真实世界中，对用户的进行身份认证的方法基本上可以分三种：
 - @**what you know**：根据你所知道的信息来证明你的身份
 - @**what you have**：根据你所拥有的东西来证明你的身份
 - @**who you are**：直接根据独一无二的身体特征来证明你的身份，比如指纹、面貌等

5

常见的电子身份认证技术手段

方法	技术
What you know	静态口令
	智能卡(IC卡)
What you have	短信口令
	动态口令牌(基于时间戳)
	动态口令牌(基于计数器)
	USB KEY(Challenge/Response模式)
	USB KEY(电子证书模式)
Who you are	生物特征验证技术 (指纹、掌纹、声纹、视网膜、虹膜、脸型、体态、DNA等)



7

电子身份认证常见应用场景

@登录操作系统

@Windows、Mac OS、Linux.....

@上网

@TUNet客户端

@无线网登录

@登录网站

@校内：INFO、网络学堂、图书馆.....

@校外：书库、论坛、微博、淘宝、电子银行.....

@玩游戏

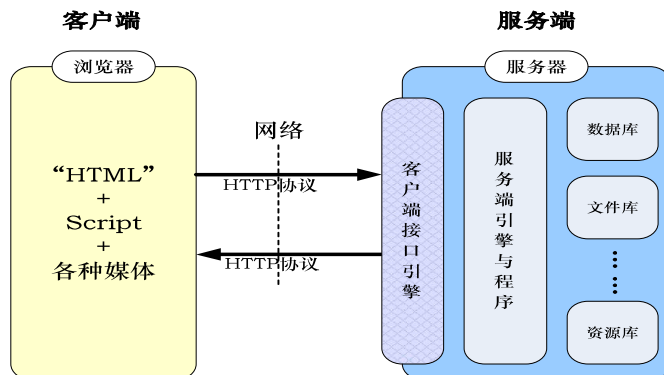
@.....

8

step by step 设计安全可靠的网站用户认证模块

10

网站系统的基本构成



12

构建安全网站的技术手段



@正面

@认证 (Authentication)

@授权 (Authority)

@审计 (Auditing)

@反面

@消灭各个层面的漏洞 (Bug)

@网站是一个复杂系统，安全问题涉及很多方面

step by step: 设计安全可靠的网站用户认证模块

9

网站系统的基本构成

@网站Web又称为万维网 (World Wide Web)，基本结构是采用开放式的客户/服务器结构 (Client/Server, C/S)

@完整的网站系统分成三个部分：

@服务器端：Web Server

@客户端：浏览器Browser、专用程序

@通信协议：HTTP协议

@网站系统中常用的技术

@HTML、JavaScript、多媒体技术

@服务端技术

11

HTTP协议

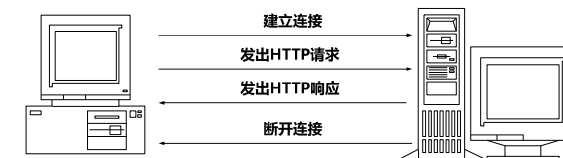
@超文本传输协议HTTP (Hyper Text Transfer Protocol) 是基于TCP/IP的应用层协议。

@HTTP的主要特点：

@典型的C/S工作方式，简单快速，数据传输较少

@可以通过HTTP协议传输任意类型的数据对象

@HTTP是面向一次连接的无状态网络协议



13

网站用户认证技术

@最简单的网站用户认证：**HTTP的Basic认证**

@改进解决Basic认证的问题：基于表单Form的身份认证

@若干网站身份增强认证技术

- @手机短信口令
- @动态口令
- @USB KEY
- @数字证书

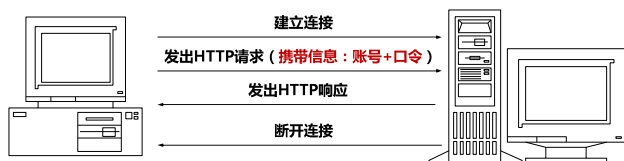
14

HTTP的Basic认证：机制原理

@用户身份凭证：账号+静态口令

@由于HTTP协议是面向一次连接的无状态网络协议，因此需要在每次发出HTTP请求时，把用户身份凭证的明文发送到服务器端，服务器与存储在服务器端的用户凭证进行**比较**

@这是HTTP的Basic认证机制

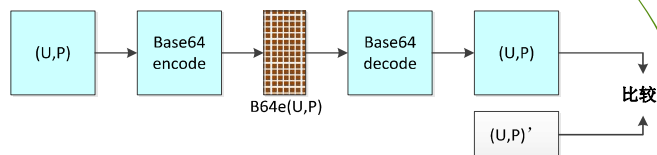


16

HTTP的Basic认证：认证过程

@对账号口令 (U,P) 进行Base64编码(便于文本方式传输)，发送到Web服务器端

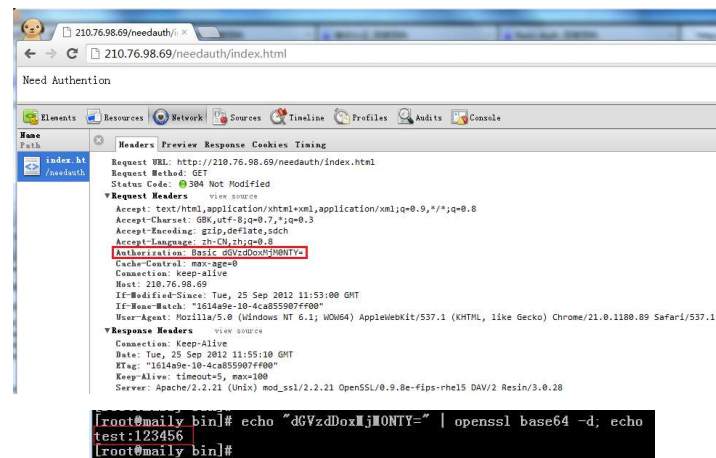
@服务器端接收后进行Base64解码得到 (U,P)，并与本地存储的账号口令 (U,P)' 进行**比较**，认证用户身份



18

最简单的网站用户认证： HTTP的Basic认证

15



17

HTTP的Basic认证：优缺点

@优点

- @没有复杂的交互过程
- @使用非常简单

@缺点

- @每次都需要传递账号和口令，易于被监听、盗取，安全性低
- @本地（浏览器）还需要保存账号和口令，存在安全隐患
- @服务器端处理每个HTTP请求都要进行一次身份验证过程，效率较低

19

HTTP的Basic认证：改进1

@解决账号口令明文传输可能被监听盗取的问题

- @使用加密技术
- @使用消息认证(MA)技术

@解决账号口令长期本地保存以及服务器端每次都进行账号口令验证的问题

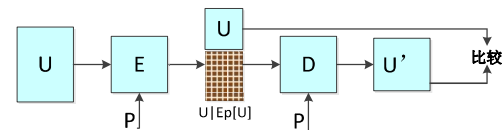
- @使用表单验证+session的机制

20

HTTP的Basic认证：改进2

@账号口令加密

- @加密需要密钥key，正好把口令作为密钥，即： $k = \text{口令}$
- @每次传输 $(U \parallel E_k[U])$ ，
 - @算法E可以是任意对称加密算法，如3DES、AES、RC5等



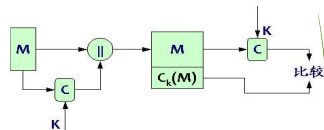
@但是，无法躲避重放攻击(Replay)！

21

HTTP的Basic认证：改进3

@使用消息认证中的MAC认证技术，

- @客户端和服务端共享密码K（口令）



@采用挑战/响应(Challenge/Response)机制，需要进行两次HTTP请求

- @第一次请求：服务器向客户端返回一个挑战码M
 - @挑战码M由服务器随机生成，可以有效避免重放攻击
- @第二次请求：客户端使用MAC认证发送MAC码，服务器端进行验证
 - @C可用hash算法，如SHA1

22

改进Basic认证的问题： 基于表单Form的身份认证

23

基于表单Form的身份认证：目标

@为了解决Basic认证的问题：

- @账号口令长期本地保存
- @服务器端每次都进行账号口令验证



24

Web的Session机制

@一个Session包括特定的客户端、特定的服务器端以及特定的操作时间段。

@Session的工作原理：

- @当某个Session首次启用时，服务器会产生一个唯一的标识符发送到客户端，客户端收到后会存储下来
 - @唯一标识符通常是一长串随机字符串，通常存于内存或本地文件中
 - @在客户端浏览器上，唯一标识符通常用Cookie技术进行存储
 - @Cookie包含：key/value值对、作用域、作用路径、有效时间
- @在Session存活期间，客户端每次向服务器发送的HTTP请求都会包含上述唯一标识符，使得服务器能够把前后多次请求关联起来
- @当Session结束时，服务器端和客户端都应当销毁上述唯一标识符

25

基于表单Form的身份认证：过程

@基于表单的Web身份认证过程通常包括三个步骤：

- @第一步：客户端向服务器发送请求，服务器返回包含表单的页面
- @第二步：用户按要求填写表单的内容完成后，客户端把表单内容发送到服务器；服务器获取表单中的内容后，进行验证，验证通过则启动Session并返回给客户端
 - @服务器端可以完成接收到客户端发送的表单数据，即：服务器端能获得用户账号口令的明文，可以直接进行比较验证
- @第三步：客户端后续的请求包含Session的唯一标识符，服务端验证唯一标识符的合法性
 - @通常基于一个以标识符为key的哈希表数据结构

26

基于表单Form的身份认证：优缺点

@优点

- @简单、方便、易用

@缺点

- @安全性低，账号和口令明文在网络上传输，很大程度上存在被监听盗取的可能

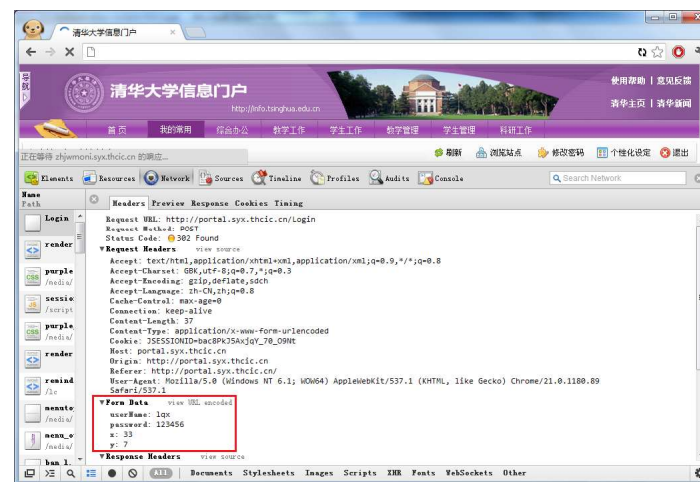
28

基于表单Form的身份认证：改进2

@在第二步传输账号口令时，
使用传输层SSL协议传输HTTP请求

- @SSL协议(Secure Sockets Layer) 是为网络通信提供安全及数据完整性的一种安全协议，SSL在传输层对网络连接进行加密，确保数据在网络上的传输过程中不会被截取、窃听和篡改
- @SSL协议可以验证客户端和服务端软件的身份，确保数据发送到正确的客户机和服务器
- @在Web中启用SSL协议只需把URL改为<https://>即可
- @SSL协议在“网络安全协议”中会有详细介绍

30



27

基于表单Form的身份认证：改进1

@引入Challenge/Response机制，避免口令明文通过网络传输，并且能避免重放攻击

- @第一步，服务器返回表单页面时，包含一个Challenge（通常为随机字符串+时间戳，记为M）
- @第二步，用户完成表单填写提交请求之前，以口令为加密密钥（ $k = \text{口令}$ ）计算 $E_k[M||U]$ ，并将账号U和 $E_k[M||U]$ 发送到服务器，服务器收到后，用存储的口令 k' 计算 $E_{k'}[M||U]$ ，与收到的 $E_k[M||U]$ 比较，完成用户的身份验证

29

常见的不安全做法

@常见的不安全做法

- @验证成功后，把账号口令以Cookie机制存放在浏览器端
- @验证成功后，把账号口令进行加密后 $E_k[U||P]$ 以Cookie机制存放在浏览器客户端
- @带来的问题：口令泄露，重放攻击
- @如果非要这么用
 - @存放内容加上时间戳t，加密 $E_k[t||U||P]$
 - @服务器端每次需要验证时间戳

31

若干网站身份增强认证技术

32

手机口令

- @通常不会单独使用手机口令，而是与静态口令配合形成双因素认证方式
- @用户建立帐号时，需要与一个手机号码绑定
- @用户登录网站后，进行某些关键操作时，会触发系统向手机发送一条包含随机验证码的短信
- @当用户正确输入该随机验证码后，才可以继续刚才的关键操作



34

基于时间同步的动态口令原理

- @服务器与令牌端基于同样的算法，且双方的时间经过初始化校准（记录时间偏移）
- @算法原理
 - @服务器与令牌端共享一个key和令牌序列号sn
 - @每次计算先对时间t按区间n划分取整（上取整或下取整）
 - @计算 $F(H(t||key||sn))$ ，其中H是散列函数，F是压缩函数，把散列结果转换为6位或8为十进制数
- @用户验证时，提交令牌端计算的数字，服务器端按同样的算法计算（考虑到网络造成的时间差可以计算t-n,t两个值）进行比较验证

36

其它网站身份增强认证

- @仅靠静态口令认证用户安全强度不足，需要更高强度的认证方式
- @通常在静态口令的基础上加上其他因素的认证方式，形成双因素认证或多因素认证方式
 - @手机短信口令
 - @动态口令
 - @USB KEY
 - @数字证书

33

动态口令

- @动态口令牌通常也与静态口令验证形成双因素身份验证模式，广泛应用于VPN、网上银行、电子商务等领域
- @动态口令牌是客户手持用来生成动态密码的终端（手机上运行的程序或者是专用硬件终端）
 - @有的基于时间同步方式，每n秒变换一次有效的动态口令
 - @有的基于计数器同步方式，每次触发计数器累进会产生一个一次有效的动态口令

35

USB KEY

- @USB KEY是一种USB接口的硬件设备，它内置单片机或智能卡芯片，可以存储用户的密钥或数字证书
- @USB KEY可以用于身份认证、数据加密和数字签名
 - @USB KEY中的私钥不可导出，仅能用于内置的加密或签名运算
- @利用USB KEY实现对用户身份认证的原理是：
 - @服务器端产生随机消息M送到USB KEY内
 - @USB KEY用内部私钥对M进行签名
 - @服务器使用用户的公钥验证签名

37

数字证书



- @数字证书就是互联网通讯中标志通讯各方身份信息的一系列数据，提供了一种在Internet上验证身份的方式
- @数字证书由权威机构，又称为证书授权（Certificate Authority）中心发行
- @数字证书是一个经证书授权中心数字签名的文件
 - ③最简单的证书包含一个拥有者的公钥、名称以及证书授权中心的数字签名
- @数字证书所有者对应的私钥应该另外妥善存储

38

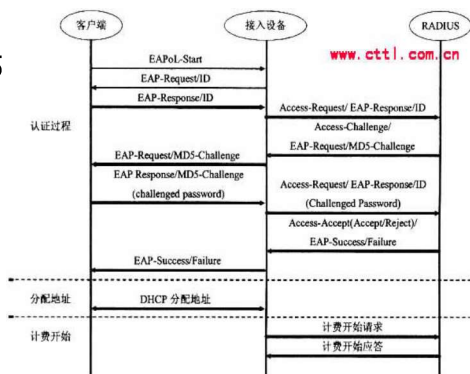
操作系统登录身份认证

- @Windows系统
 - ③用户帐号信息存在SAM数据库中(Security Account Manager)
 - ③用户口令会进行加密处理存放到SAM中
 - ③用户输入帐号口令后，系统对输入的信息进行加密处理，与SAM中的数据进行**比较**，成功则完成后续登录过程
- @Linux操作系统
 - ③用户帐号信息通常存放于/etc/passwd和/etc/shadow文件中
 - ③用户口令经过散列处理后进行存储，形如
test:\$1\$ZcUVmoZK\$gG/3O0k.Qqjt4UY4ks9UF/:12766:0:99999:7:::
 - ③用户输入帐号口令后，系统对输入的用户口令进行同样的散列处理，与shadow文件中的信息进行**比较**，成功则完成后续登录过程

40

802.1x认证过程

- ⑧ 802.1X主要是基于MD5的Challenge/Response 认证



42

其它身份认证应用

39

802.1X网络准入身份认证

- @802.1X是一个认证协议，是一种基于端口对用户进行认证的方法和策略
- @对于一个端口：
 - ③如果认证成功，就“打开”这个端口，允许所有的网络包通过
 - ③如果认证不成功，就使这个端口保持“关闭”，此时只允许802.1X的认证数据包通过

41

TUNet用户登录认证

- @TUNet是学校用于登录校园网的客户端软件
- @TUNet登录认证原理
 - ③客户端C向服务器S发起TCP连接
 - ③连接成功后，C向S发送“Hello”请求
 - ③S产生8字节随机数R，发送给C
 - ③C计算 $M = E_k(R)$ ，其中E为3DES算法， $k = MD5(\text{password})$ ，把M发送给S
 - ③S同样计算得到M，与C发送过来的比较
- @服务器每次发送给客户端的响应都包含一个新的8字节随机数 R_n ，下一次请求需要包含 $H(M || R_n)$ 以进行验证

43

SSH协议登录认证



@SSH是安全登录协议，替换原来的TELNET协议

@SSH全程通信数据加密

@SSH协议基于Diffie-Hellman密钥交换过程，并在第一次密钥交换之后记录服务器的Diffie-Hellman参数(q,a)，形如

```
@ 192.168.1.1 ssh-rsa
AAAAB3NzaC1yc2EAAAABIwAAAIEA7ehjuoPsPeNH0htxIE2okxIHP1D8wsRqnca/7xhM
nUv3V0fzRS/ktKgclFc9JIOLR3iWkZTPfJyepVfP9mJT4Nr77NJREBYI42W4IPpbhn6U6E
b259g/gecqADzmbhdl6AOD7C0Qj5i/UU7rpK0pKSrhv/5AATER9KgeELwU60=
```

@SSH支持对服务器和客户端身份真实性进行验证

课程主要内容

- @一、计算机网络安全概论
- @二、密码学基础
 - @密码学的基础知识
 - @对称密钥密码算法
 - @公开密钥密码算法
 - @密钥的分配
- @三、认证技术
 - @消息认证技术
 - @身份认证及其实例
- @四、网络安全协议
 - @网络层安全：IPsec、IKE
 - @传输层安全：SSL
 - @应用层安全：电子邮件安全、电子商务安全、WEB安全
- @五、各类专用技术

2

安全通信协议概述

4

IP级安全性



@IPsec保障了IP级的安全性，包括：

- @认证：
确保收到的包是从包头标识的源端发出的，而且该包在传输过程中未被篡改。
- @保密：
将报文加密后传输，防止第三方窃听。
- @密钥管理：
密钥管理机制与密钥的安全交换相关。

6

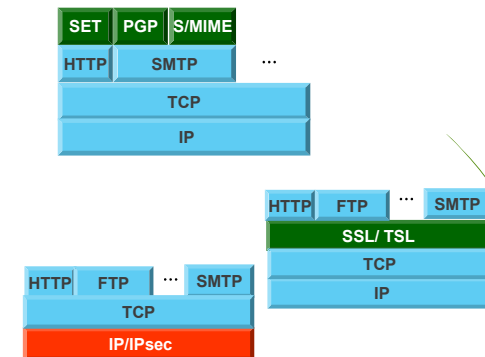
网络层安全协议

- @安全通信协议概述
- @IPSec 概述
 - @历史、应用、实施
- @IPSec 体系结构
 - @文档、服务、安全关联、传输模式和隧道模式
- @认证头AH (Authentication Header)
- @封装安全载荷ESP (Encapsulating Security Payload)
- @安全关联组合

3

安全通信协议

- @应用层
 - @S/MIME
 - @PGP
 - @SET
- @传输层
 - @SSL
 - @TLS
- @网络层
 - @IPSec



5

IPSec 概述

历史、应用、实施

7

现有IP协议的安全特性

@无连接，不保证顺序到达。
重复包、丢失包，设备简单、无状态。

@所提供的安全服务

- @认证：无
- @完整性：无
- @保密：无
- @访问控制：基于IP地址，但是不完备

@所面临的威胁

- @窃听、伪造IP地址、篡改、重发

8

IPSec的历史

@1994年，互联网体系结构委员会IAB 发布了RFC1636---
“Internet 体系结构中的安全性”，明确指出大多数人认为互联网需要更多更好的安全性，并指明使用密钥领域的安全机制。

@IAB认为应该在网络层实现端到端的安全性

- @数据源认证机制
- @数据加密机制
- @密钥管理

@IPSec的原理在于可以在IP层加密和/或认证所有流量。
这些安全能力在IPv4和IPv6中都适用。

10

IPSec 的应用

@IPSec的用途包括：

@远程安全访问互联网

- @使用了IP安全协议的终端用户可以通过本地调用访问互联网服务提供商ISP，以获得对公司网络的安全访问，减少了雇员上班和远程通信的费用

@与合作者建立外联网和内联网联系

- @IPSec可以与其它组织进行安全通信，确保认证、保密和提供密钥交换机制。

@加强电子商务安全性

- @虽然一些WEB和电子商务应用软件是建立在安全协议之上的，但使用IPSec能够增强安全性。

12

IP Packet

@IP 协议是无状态、无连接。

@IP数据报由20个字节的报头和来自传输层的数据构成。

0	4	8	16	19	24	31
VERS	HLEN	Service Type	Total Length			
Identification			Flags	Fragment Offset		
Time to Live		Protocol	Header Checksum			
Source IP Address						
Destination IP Address						
IP Options (If Any)					Padding	
Data						
...						

9

IPSec 的应用

@IPSec提供了在LAN、WAN和互联网中安全通信的能力。

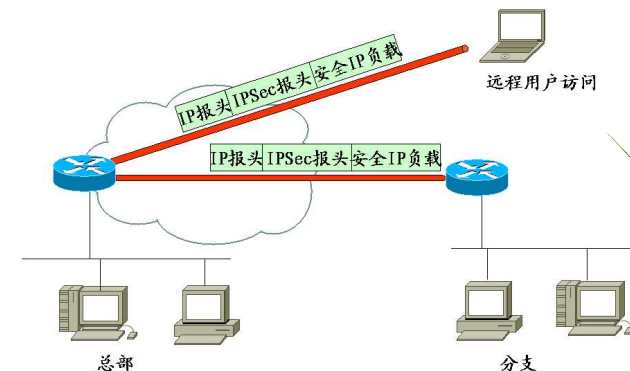
@IPSec的用途包括：

@分支机构通过互联网安全互联

- @一个公司可以在互联网和公网WAN上面建立安全的虚拟专用网，使得依赖于互联网的交易成为可能，并减少了对专用网络的需求，节省了开销和网络管理费用。

11

虚拟专用网VPN



13

IPSec的实施

@在主机上实现（与操作系统集成）

- @保障端到端的安全
- @能够实现所有模式
- @能够针对用户的每个会话提供安全保障
- @对应用透明，不必修改用户或服务器系统的软件
- @IPSec可以对最终用户透明



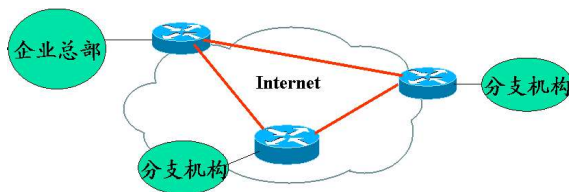
14

IPSec的实施



@在路由器上实施

- @虚拟专用网(VPN): 可以对通过公用网络在两个子网之间流动的数据提供安全保护。
- @虚拟专用拨号网(VPDN), 安全的远程访问。



16

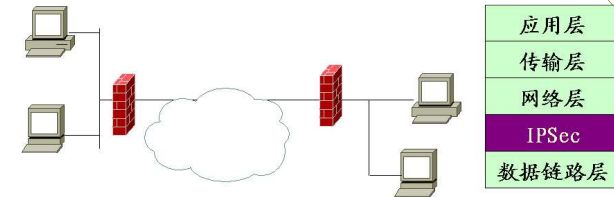
IPsec文档

18

IPSec的实施

@在防火墙上实施

- @无须改变操作系统
- @为内部所有的应用提供安全服务



15

IPSec 体系结构

文档、服务、安全关联、传输模式和隧道模式

17

IPSec文档

@IPSec有很多文档，重要的有1998年发布的RFC 2401/2402/2406/2408。

- @RFC 2401: 安全结构概述
- @RFC 2402: IP扩展的包认证描述 (IPv4和IPv6)
- @RFC 2406: IP扩展的包加密描述 (IPv4和IPv6)
- @RFC 2408: 特定加密机制

@IPv6必须支持这些特性，IPv4可选。

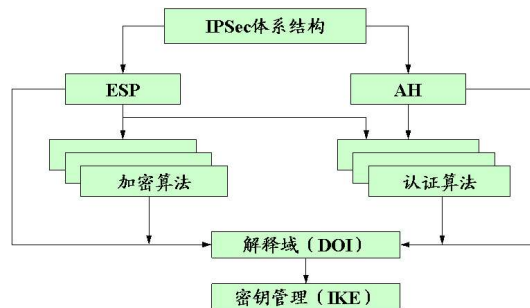
这两种情况都是在主IP报头中使用了扩展报头实现安全特性。

- @认证的扩展报头称为认证头，
- 加密的扩展报头称为载荷安全封装头。

19

IPSec文档

@除了上述四个文档，IETF下设的IP安全协议工作组又做了大量的工作，整个IPSec 文档被分成七个部分。



20

IPSec文档

@整个IPSec 文档被分成七个部分：

@认证算法

- @一系列描述AH中使用的各种认证算法和ESP认证选项的文档。
- @RFC2403, The Use of HMAC-MD5-96 within ESP and AH.
- @RFC2404, The Use of HMAC-SHA-1-96 within ESP and AH.

@密钥管理

- @描述密钥管理模式的文档。

@解释域 (DOI)

- @其他文档需要的为了彼此间互相联系的一些值，包括经过检验的加密和认证算法的标识以及操作参数，比如密钥的生存期。

22

IPSec提供的服务

@IPSec在IP层提供安全服务，系统可以选择所需要的安全协议，确定该服务所用的算法，并提供安全服务所需加密密钥。

@使用AH协议报头的认证协议和为数据包设计的加密/认证协议ESP，都提供安全性。

@ESP可以分成支持和不支持认证两种。

@AH和ESP均支持基于分布密钥的访问控制，而流量管理则与加密协议相关。

24

IPSec文档

@整个IPSec 文档被分成七个部分：

@体系结构：

- @定义IPSec技术的一般性概念、需求和机制。

@认证头AH：

- @IP 数据包中的一个扩展域，用于提供数据的源发认证和完整性保护。

@封装安全载荷ESP：

- @IP 数据包中的一个扩展域，用于提供数据保密、源发认证和完整性保护。

@加密算法

- @一系列描述ESP中使用的各种加密算法的文档。
- @RFC2405, The ESP DES-CBC Cipher Algorithm With Explicit IV.

21

IPsec提供的服务

23

IPSec提供的服务

@它们提供的安全服务包括：

@访问控制

@无连接完整性

@数据源发认证

@拒绝重放数据包

@保密性（加密）

@有限的信息流 保密性

	AH	ESP(只加密)	ESP(加密并认证)
访问控制	✓	✓	✓
无连接的完整性	✓		✓
数据源发认证	✓		✓
检测重放攻击	✓	✓	✓
机密性		✓	✓
有限的通信流保密		✓	✓

25

安全关联SA

26

安全关联

@一个安全关联SA由三个参数唯一确定：

@安全参数索引SPI(Security Parameters Index)

@SPI是一个和SA相关的位串，仅在本地有意义。

@SPI由AH和ESP携带，使得接收方能够选择合适的SA处理接收包。

@IP 目的地址IPDA

@只允许使用单一地址，表示SA的目的地址，可以是用户末端系统、防火墙或者路由器。

@安全协议标识

@标识该关联是一个AH安全关联或ESP安全关联。

28

SA的参数

@SA的参数包括

@序数计数器

@一个32位用于生成AH或ESP头中的序数值（必须实现）

@序号溢出标志

@表明序列号计数器溢出是否应该产生审查事件并且阻止SA上额外包的传输的标志（必须实现）

@反重放窗口：

@用于决定输入AH或ESP报文是否是重播的32位计数器（必须实现）

@AH信息

@认证算法，密钥，密钥生存期和AH的相关参数（AH必须实现）

30

安全关联

@在IP的认证和保密机制中的一个核心概念是安全关联SA（Security Association）。

@安全关联SA 是IPsec通信双方之间对某些要素的一种协商，一组安全信息参数集合。包括：协议、操作模式、密码算法、认证算法、密钥、密钥生存期等。

@关联是发送方和接收方之间的单向关系，该关联为双方的通信提供了安全服务。

@如果需要双方安全交换，则建立两个安全关联。

@安全服务可由AH或者ESP提供，但不能两者都提供。

27

安全关联

@在任何IPSec实现中，都有一个安全关联数据库SADB，它定义了与每个SA相关联的参数。

A(1.1.1.1) ←→ B(2.2.2.2)

source	dst	protocol	spi	SA记录
1.1.1.1	2.2.2.2	AH	11	MD5, K1
2.2.2.2	1.1.1.1	ESP	12	DES, K2
2.2.2.2	1.1.1.1	AH	13	SHA, K3

←→ ←→ ←→

A的 SADB

29

SA的参数

@SA的参数包括

@ESP信息

@加密和认证算法，密钥，初始值，密钥生存期和ESP的相关参数（ESP必须实现）

@SA的生存期：

@一个时间间隔或字节计数。

生存期结束时，SA必须终止，或用一个新的SA替换。

@IPSec协议模式：

@隧道，传输（必须实现）

@Path MTU：

@任何遵从的最大传送单位路径和迟滞变量。（必须实现）

31

安全关联

- @由安全关联数据库SADB提供的功能必须在IPSec的任何实现中都存在，但提供功能的方式由实现者决定。
- @认证和保密机制与密钥管理机制相互独立。分配密钥使用的密钥管理机制只通过安全参数索引SPI与认证和保密机制相联系。
- @在IP包中，安全关联由IPv4或IPv6报头中的目的地址唯一标识，SPI被封装在AH或者ESP扩展头中。

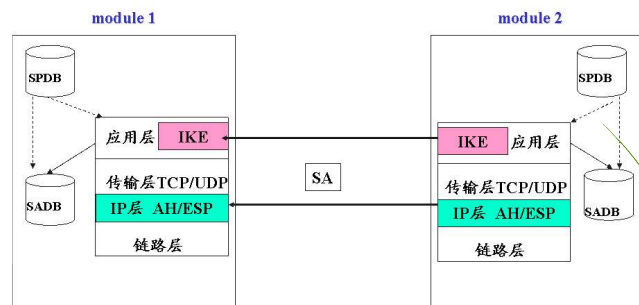
32

SA选择子

- @每个SPDB入口由一个IP集合和上层协议定义，称为选择子。实际上，这些选择子用于过滤输出流量，并将它们映射到某个特定的SA。
- @每个IP包的输出过程如下：
 - @在SPDB中比较包相应域的值，寻找匹配的SPDB入口，可能是零个或者多个。
 - @如果存在SA，则选定SA和其关联的SPI。
 - @执行所需的IPSec处理（AH或ESP）。

34

工作过程



36

SA选择子

- @IPSec提供了多种方式在IP通信中实现IPSec服务，用户可根据自己的意愿进行配置。IPSec对需要IPSec保护的流量和不需要IPSec保护的流量进行了大粒度区分。
- @IP流量与特定SA相关是通过安全策略数据库SPDB的。SPDB至少应该包括定义IP流量子集的入口、指向该流量SA的指针。
- @更复杂的情况：多个入口可以与一个SA相连，或者多个SA与一个SPDB入口相连。

33

SA选择子

- @SPDB入口由下列选择子确定
 - @目的IP地址、源IP地址
 - @用户标识
 - @数据敏感性级别
 - @传输层协议
 - @IPSec协议
 - @源端口和目的端口
 - @IPv6报类
 - @IPv6报流标签
 - @IPv6报服务类型

35

传输模式和隧道模式

37

传输模式和隧道模式

@AH和ESP均支持两种模式：传输模式和隧道模式。

@传输模式(Transport Mode)主要为上层协议提供保护，同时增加了IP包载荷的保护。

- @典型的传输模式用于两台主机之间进行的端到端通信。
- @传输模式的ESP加密和认证(可选)IP载荷，不包括报头。
- @传输模式的AH认证IP载荷和报头的选中部分。

38

隧道模式

@隧道模式(Tunnel Mode)对整个IP包提供保护。

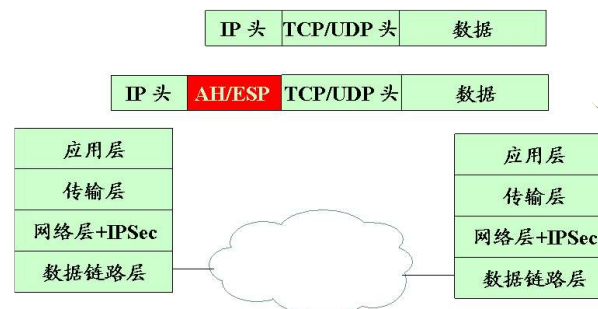
- @当IP包加上AH/ESP域后，整个数据包和安全域被当作一个新的IP载荷，并拥有一个新外部IP报头。新的IP数据包利用隧道在网络中传输，途中的路由器不能检查内部IP报头。
- @ESP在隧道模式中加密和认证(可选)整个内部IP包，包括内部IP报头。
- @AH在隧道模式中认证整个内部IP包和外部IP报头的选中部分。

40

认证头AH

42

传输模式

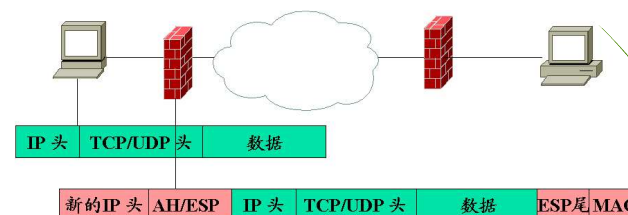


39

隧道模式

@隧道模式 (Tunnel Mode)

@Firewall or Router



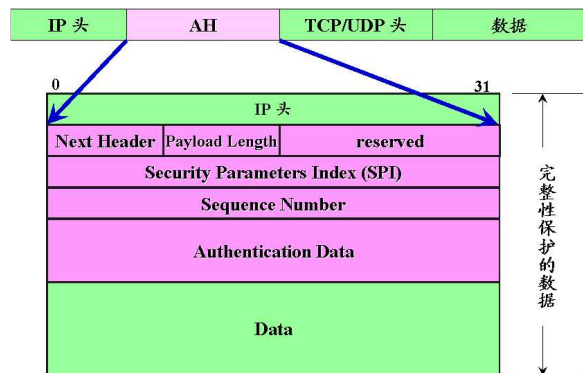
41

认证头Authentication Header

- @认证头支持数据完整性和IP包的认证。
- @数据完整性确保在包的传输过程中内容不可更改。
- @认证确保末端系统或者网络设备对用户或者应用程序进行认证，并提供相应的流量过滤功能，同时还能防止地址欺诈攻击和重放攻击。
- @认证基于消息认证码MAC，双方必须共享一个公钥。

43

认证头的组成



44

认证头的组成

@Security Parameters Index (安全参数索引)

@SPI为数据报识别安全联合的 32 位伪随机值，SPI=0被保留，表明“没有安全关联存在”。

@Sequence Number(序列号)

@32bits，单调递增计数器，用于防范重放类型攻击。

@Authentication Data(认证数据AD)

@变长域，必须是32 bits的整数倍。

@包含完整性校验值ICV或者包的MAC。

46

反重放攻击

@由于IP是无连接、不可靠的服务，协议不能保证包按照顺序传输，也不能保证所有的包均被传输。IPsec认证文档中声明，接收方应该实现一个大小为W的窗口（W的默认值为64）。

@窗口的右边界代表最大的序号N，记录目前收到的合法包的最大序列号，任何序列号在N-W+1到N之间的包均可以被正确接收，并标记窗口的正确位置。

48

认证头的组成

@Next header(邻接头)

@8 bits，标识数据载荷中的封装方式或协议。例如是TCP/UDP/ICMP、AH、ESP。

@Payload length(有效载荷长度)

@8 bits，以 32 位字为单位的认证数据字段长度。

@Reserved(保留字)

@16 bits，保留以供将来使用。

@发送时必须设置为全零。这个值包含在认证数据计算中，否则被接收方忽略。

45

反重放攻击

@重放攻击是指攻击者在得到一个经过认证的包后，又将其传送到目的站点的行为。

@重复接收经过认证的IP包可能会以某种方式中断服务或者产生不可预料后果，序列号域就可以防止重放攻击。

@当建立了一个新的SA时，发送方将序列号初值设为0，每次在SA上发送一个包，则计数器加1并将值放入序列号域，则使用的第一个值就是1。

@发送方不允许循环计数，否则，同一个序列号就可以产生多个合法的包。如果该序列号到达 $2^{32}-1$ ，则SA必须中止，用新的密钥协商声称新的SA。

47

反重放攻击

@包到达之后：

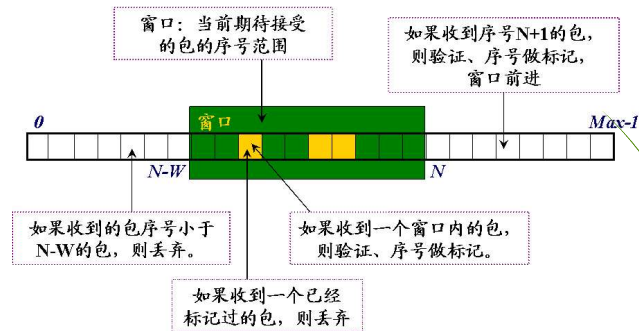
@如果所接收的包在窗口中而且是新包，则验证MAC。如果通过验证，则在窗口中标记相应的位置。

@如果所接收的包超过窗口右边界而且是新包，则验证MAC。如果通过验证，则以这个序列号为窗口的右边界并在窗口中标记相应的位置。

@如果所接收的包超过窗口的左边界或未通过验证，则丢弃此包，并生成审核事件。

49

反重放攻击



50

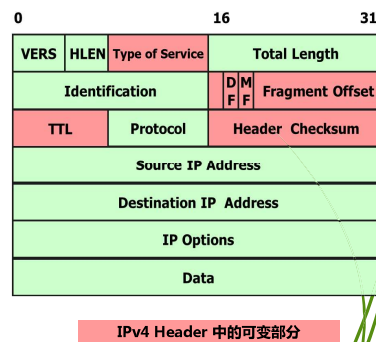
完整性校验值

@在以下字段计算MAC值:

@IP报头: 传输过程中不变的部分和AH SA终点可以预测的部分参与计算MAC。
对于可变部分和不可预测的部分全部置0, 便于在源端和目的端计算。

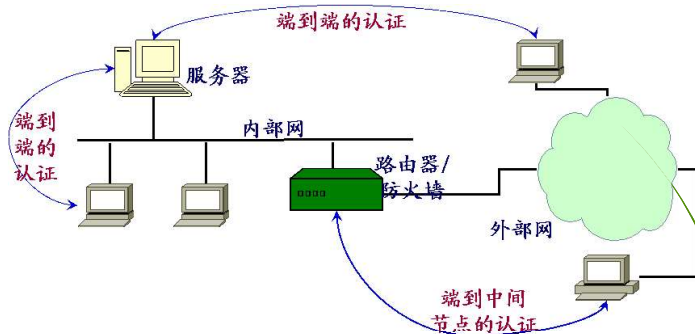
@AH报头中不计算认证数据域。认证数据域被置成0, 便于在源端和目的端计算。

@整个上层协议数据, 如TCP/UDP数据, 假设在传输过程中不变。



52

传输模式和隧道模式



54

完整性校验值

@认证数据AD域包含完整性校验值ICV。

@ICV是一种报文认证编码MAC或者MAC算法生成的截断代码。

@下面两种规范描述了ICV的生成。两者均先计算全部的HMAC值, 然后截取计算结果的前96bits。

@HMAC-MD5-96

@HMAC-SHA-1-96

51

传输模式和隧道模式

@有两种途径使用IPSec认证服务。

@直接在服务器和客户工作站之间提供认证。客户工作站可以与服务器在同一个网络中, 也可以在外部网络中。只要客户工作站和服务器的共享一个受保护的密钥, 认证就是安全的。这种模式使用的是传输模式SA。

@在服务器不支持认证的情况下, 远程工作站向防火墙证明自己身份, 以便访问整个内部网络的时候, 使用隧道模式SA。

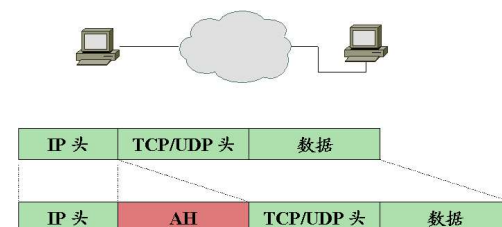
53

传输模式

@传输模式

@通常以端到端方式实现(在主机上实现)。

@不修改IP头, 只添加AH头。



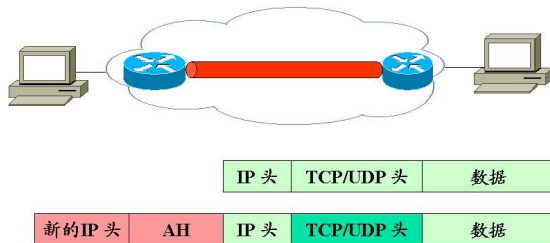
55

隧道模式



@隧道模式

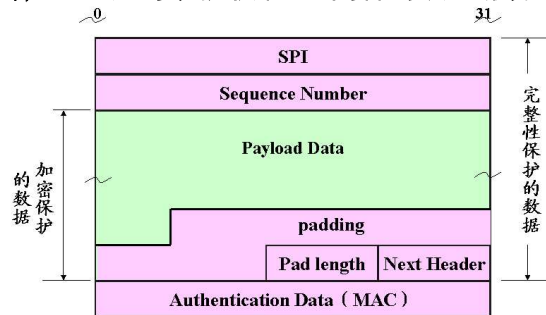
- @通常在防火墙或路由器上实现。
- @把整个IP包作为数据，增加一个新的IP头、AH头。



56

封装安全载荷ESP

- @封装安全载荷ESP提供保密性服务包括报文内容保密和流量限制保密，ESP还可以提供和AH同样的认证服务。



58

ESP格式

@Padding(填充域)

- @可变长域，范围在0~255字节。

@Pad Length: 填充域长度。

@Next header(邻接头)

- @8 bits，标识数据载荷中的封装方式或协议。例如是TCP/UDP/ICMP、AH、ESP。

@Authentication Data (认证数据AD)

- @变长域，必须是32 bits的整数倍。
- @包含完整性校验值ICV或者包的MAC。

60

封装安全载荷ESP

57

ESP格式

@Security Parameters Index(安全参数索引)

- @SPI为数据报识别安全联合的32位伪随机值，SPI=0被保留，表明“没有安全关联存在”。

@Sequence Number(序列号)

- @32bits，单调递增的计数器，用于防范重放类型的攻击。

@Payload length(有效载荷长度)

- @8 bits，以32位字为单位的认证数据字段长度。

59

ESP支持的加密与认证算法

- @载荷数据、填充数据、填充长度和邻接头域都在ESP中被加密。可用加密算法有：

- @3DES
- @RC5
- @IDEA
- @CAST
- @Blowfish

- @与AH相同，ESP支持使用默认为96位的MAC，并且支持HMAC-MD5-96和HMAC-SHA-1-96。

61

填充

@填充域的功能如下：

- @如果加密算法需要明文是某个数目字节的倍数，则填充域可以用于扩展明文长度。
- @ESP格式需要填充长度和邻接头域为右对齐的32位字，填充域用来保证这种排列。
- @增加额外的填充域可以隐藏载荷的实际长度，并提供部分流量保护。

62

传输模式

@加密和认证(可选)直接由两个主机提供。

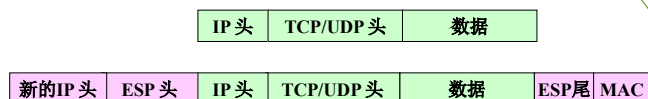
- @在源端，包括ESP尾和整个传输层分段的数据块被加密，块中的明文被密文替代，形成要传输的IP包。如果选择了认证，则加上认证。
- @将包送往目的地。中间路由器需要检查和处理IP报头以及任何附加的IP扩展头，但不需要检查密文。
- @目的节点对IP报头和任何附加的IP扩展报头进行处理后，利用ESP报头中的SPI解密包的剩余部分，恢复传输层分段数据。

64

隧道模式

@隧道模式的ESP用于加密整个IP包。在此模式中，将ESP头作为包的前缀，并在包后附加ESP尾，然后对其进行加密。该模式用于流量计数分析。

@隧道模式可以创建虚拟专用网。



66

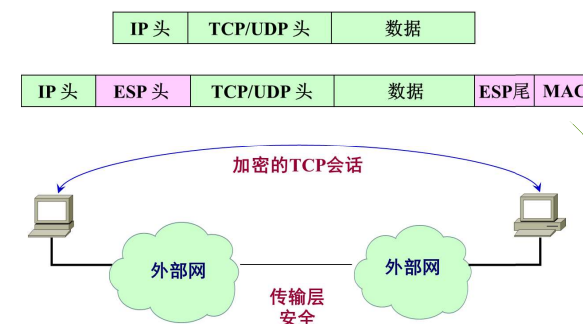
传输模式和隧道模式

@IPSec的ESP提供两种使用途径。

- @加密和认证（可选）直接由两个主机提供。采用传输模式。
- @隧道模式的ESP用于加密整个IP包。隧道模式可以创建虚拟专用网。

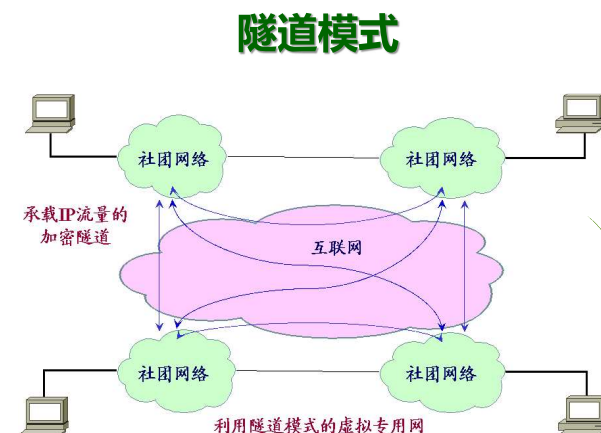
63

传输模式



65

隧道模式



67

安全关联组合

68

安全关联组合

@安全关联可以通过两种方式组合成束。

@传输邻接：在不使用隧道的情况下，对一个IP包使用多个安全协议。组合AH和ESP的方法仅允许一级组合。

@隧道迭代：指通过IP隧道应用多层安全协议。由于每个隧道可以在路径上的不同IPSec节点起始和结束，因此，该方法允许多层嵌套。

70

安全关联组合

@单个SA可以实现AH或者ESP，但是不能两者都实现。

@有的时候，特定流量需要在主机间提供IPSec服务，并在安全网关间(例如防火墙等)为相同流量提供分离的服务。为了达到理想的IPSec服务，需要为相同流量提供多个SA。

@安全关联束是指提供特定的IPSec服务集所需的一个SA序列。安全关联束中的SA可以在不同节点中止，也可以在不同节点中止。

69

AH和ESP的典型组合



Transport

Tunnel

1. [IP1][AH][upper]
2. [IP1][ESP][upper]
3. [IP1][AH][ESP][upper]

4. [IP2][AH][IP1][upper]
5. [IP2][ESP][IP1][upper]

@这里upper指上层协议数据

@IP1指原来的IP头

@IP2指封装之后的IP头

71

课程主要内容

- @一、计算机网络安全概论
- @二、密码学基础
 - @密码学的基础知识
 - @对称密钥密码算法
 - @公开密钥密码算法
 - @密钥的分配
- @三、认证技术
 - @消息认证技术
 - @身份认证及其实例
- @四、网络安全协议
 - @网络层安全：IPsec、IKE
 - @传输层安全：SSL
 - @应用层安全：电子邮件安全、电子商务安全、WEB安全
- @五、各类专用技术

2

密钥管理概述

4

密钥管理

- @安全策略数据库 (SPDB)决定对流入和流出的哪些数据包进行安全操作(认证或加密/解密)。
- @IPSec的安全通信之前必须建立安全关联SA。SA存储在本地的安全关联数据库(SADB)中，SA决定如何对这些包进行安全操作。

6

网络层安全协议

- @安全通信协议概述
- @IPSec 概述
 - @历史、应用、实施
- @IPSec 体系结构
 - @文档、服务、安全关联、传输模式和隧道模式
- @认证头AH
- @封装安全载荷ESP
- @安全关联组合
- @互联网密钥交换协议IKE
 - @密钥管理概述
 - @互联网密钥交换协议IKE
 - @IKE的报文格式
 - @IKE的体系结构
 - @IKE的工作模式
 - @IKE的工作过程
 - @IKE协议总结

3

安全策略Policy

- @安全策略决定了为哪种类型的包提供何种安全服务，安全策略数据库SPDB(Security Policy Database)是把IP信息流与SA联系的手段。
- @每一个SPDB表项通过一组选择子(Selector)来定义。以下的选择符确定SPDB入口：
 - @协议 (TCP/UDP/ICMP)
 - @源地址，目标地址
 - @源端口，目标端口
 - @用户ID/数据敏感性级别/服务类型

5

SPDB 和 SADB

- @安全策略数据库和安全关联数据库

A(1.1.1.1) ←→ B(2.2.2.2)

source	dest	protocol	port	policy
1.1.1.1	2.2.2.2	TCP	80	AH
1.1.1.1	3.3.3.3	TCP	25	ESP

A的SPDB

source	dest	protocol	spi	SA记录
1.1.1.1	2.2.2.2	AH	11	MD5, K1,...
1.1.1.1	2.2.2.2	ESP	12	DES, K2,...
2.2.2.2	1.1.1.1	AH	13	DES, K3,...

A的 SADB

7

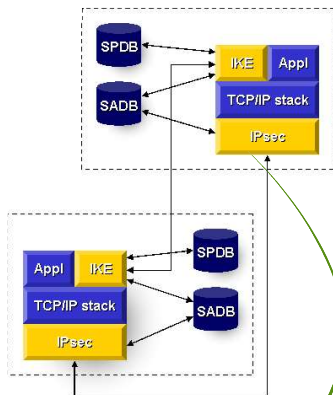
密钥管理



@IPSec的体系结构支持两种密钥管理类型：

- ①手工的：系统管理员手动地为每个系统配置自己的密钥和其它通信系统的密钥。应用于相对小规模、相对静止的环境。
- ②自动的：自动系统可在大型分布系统中使用可变配置的SA动态按照需求创建密钥。

@IKE(Internet Key Exchange)为IPSec提供了自动协商交换密钥、建立安全联盟的服务，能够简化IPSec的使用和管理



8

互联网密钥交换协议—IKE

@作为互联网的密钥交换协议，**IKE协议解决了在不安全的网络环境中安全地建立或更新共享密钥的问题。**

@IKE是非常通用的协议，不仅可为IPSec协商安全关联，而且可以为SNMPv3、RIPv2、OSPFv2等要求保密的协议协商安全参数。

@目前IKE协议只在IPSec协议中得到了应用：例如Cisco 路由器、Windows 操作系统、FreeBSD操作系统中都实现和部署了IKE。

10

IKE和IPsec

@在IPSec协议应用中，当应用环境的规模较小时，可以用手工配置安全关联SA。

@当应用环境规模较大、参与节点位置不固定时，必须有一种能够自动协商安全关联SA(各种安全参数组合，如加密算法、认证算法、传输协议、工作模式、生存时间等等)的密钥交换协议。

@IKE 协议就充当了这个角色：

IKE不但可自动地为参与通信的实体协商安全关联SA，还可以维护安全关联数据库SADB。

12

IKE Internet Key Exchange

报文格式、体系结构

工作模式、工作过程

9

IKE和IPsec

@IKE为IPSec提供了自动协商交换密钥、建立安全联盟的服务，能够简化IPSec的使用和管理。

@IKE的精髓在于：**永远不在不安全的网络上直接传送密钥**，而是通过一系列的数据交换，通信双方最终计算出共享密钥。

①即使黑客截获了双方用于计算密钥的所有交换数据，也不足以计算出真正的密钥，其核心技术就是**DH密钥交换算法**，使得IKE具备了完善的前向安全性。

②完善的前向安全性PFS(Perfect Forward Secrecy) 是一种安全特性，指一个密钥被破解，并不影响其他密钥的安全性，因为这些密钥间没有派生关系。

11

IKE和IPsec

@IKE为IPSec提供了如下功能：

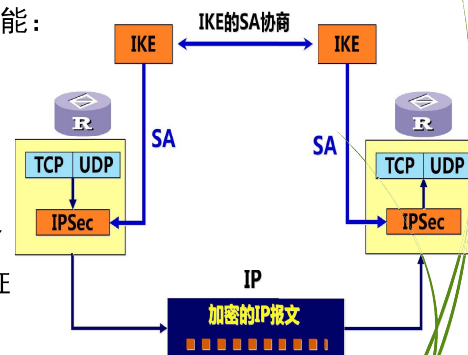
①降低手工配置复杂度

②安全关联SA定时更新

③密钥定时更新

④允许IPSec提供反重放服务

⑤允许在端与端之间动态认证



13

互联网密钥交换协议—IKE

@IKEv1的标准包括 RFC 2407, RFC 2408和RFC 2409。
IKEv2的标准是 RFC 4306。

@IKE是一个混合协议，它沿用了ISAKMP的基础、Oakley的模式以及SKEME的共享和密钥更新技术，还定义了两种密钥交换方式。

@IKE==“ISAKMP格式+Oakley模式+SKEME密钥交换”

- ③借鉴了ISAKMP协议的格式和“阶段”概念，实现了部分子集
- ③借鉴了Oakley协议“模式”概念，实现了部分子集
- ③只使用了SKEME协议中用于验证公钥加密的方法

14

IKE的报文格式

@IKE密钥交换报文格式是由ISAKMP定义的，所以也称为ISAKMP报文。

@ISAKMP可以在任何传输层协议(UDP、TCP)或IP层上实现

- ③通常ISAKMP报文是利用UDP协议的端口500进行传输

@ISAKMP双方交换的信息以“报文头+载荷”的形式传输

- ③每个ISAKMP报文由一个定长的报文头和不定数量的载荷组成。

16

ISAKMP报文头

@Flags (8 bits): Flags每一个标志位都代表密钥交换的一个特定属性。

- ③目前只定义了最低三位，其余位必须被置0

@第1位是加密位(Encryption Bit)

- ③置1表明ISAKMP消息头后面的载荷是被加密的，置0则表明是明文传输。

@第2位是约束位(Commit Bit) 用来同步密钥交换。

- ③如果密钥协商的一方A生成的ISAKMP报文将该位置1，那么对方B在安全关联建立起来后，仍要等待A发送一个安全关联建立成功的通知，才可以利用该安全关联进行保密的数据通信。

@第3位是认证位(Authentication Only Bit)

- ③置1表明该ISAKMP消息头后面的载荷是有认证的
- ③置0表明该ISAKMP报文没有做加密处理，只是做了认证处理

18

IKE的报文格式

15

ISAKMP报文头

@Initiator Cookie (32 bits):
发起者Cookie。

- ③Cookie是一个随机数，
发起者和应答者的Cookie
一起可以唯一标识一个密钥交换会话和该会话生成的IKE SA。

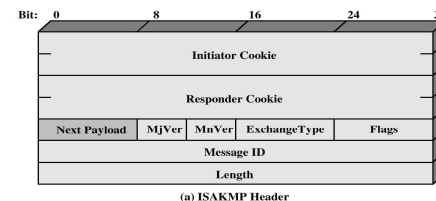
@Responder Cookie (8 bits): 密钥交换应答者的Cookie。

@Next Payload (8 bits): ISAKMP报文的第一个载荷类型。

@MjVer (4 bits): ISAKMP协议的主版本号。

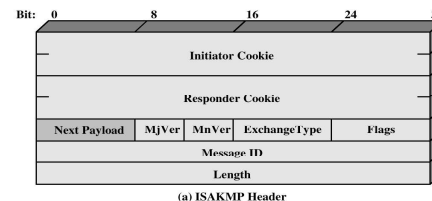
@MnVer (4 bits): ISAKMP协议的次版本号。

@Exchange Type (8 bits): 密钥交换的类型。



17

ISAKMP报文头



@Message ID (32 bits): 由第二阶段密钥协商的发起者生成的随机数，用来标志一个第二阶段密钥协商会话。

@Total Message Length (32 bits): ISAKMP报文的总长度，即报文头
和所有载荷的总长度。加密可能会导致报文长度增大。

19

ISAKMP的13种载荷

@ISAKMP定义了13种载荷，具有相同格式的载荷头。

- @Next Payload (8 bits): 该载荷的后继载荷类型。如果当前载荷最后一个，则该域被置为0
- @Reserved (8 bits): 保留字节，置0
- @Payload Length: 载荷长度（以字节为单位），该长度包括载荷头的长度。



20

ISAKMP的13种载荷



- @类型编号8: Hash载荷，传送Hash函数的结果。
- @类型编号9: Signature签名载荷，传送数字签名信息。
 - @可以用来对数据完整性进行检查，或者提供不可抵赖服务。
- @类型编号10: Nonce载荷，传送被称为nonce值的大随机数，nonce值可以防止重放攻击。
- @类型编号11: Notification通知载荷，通知对方某些信息，比如报文格式出错，SA生成等。
- @类型编号12: Delete删除载荷，通知对方删除某个或多个SA。
- @类型编号13: Vendor ID供应商载荷，提供了一种扩展手段。
 - @密钥交换双方可通过对方的Vendor ID来判断是否可以采用某种扩展。

22

IKE的两个阶段

- @IKE使用了两个阶段的ISAKMP框架。
- @第一阶段，协商创建一个通信信道（IKE SA），并对该信道进行验证，为双方进一步的IKE通信提供机密性、消息完整性以及消息源验证服务。
 - @主模式：6个消息交互
 - @积极模式：3个消息交互
- @第二阶段，使用已建立的IKE SA建立IPsec SA。
 - @快速模式：3个消息交互

24

ISAKMP的13种载荷

- @类型编号1: SA载荷，用来协商安全属性，指明解释域和状态。
- @类型编号2: Proposal载荷，包含在SA载荷中。
- @类型编号3: Transform载荷，总是包含在Proposal载荷内。
- @类型编号4: Key Exchange载荷，传送密钥的各类信息。
 - @可为Oakley, DH等密钥交换算法所用。
- @类型编号5: Identification身份认证载荷，传送身份信息。
- @类型编号6: Certificate证书认证载荷，传送证书和相关信息。
- @类型编号7: Certificate Request证书请求载荷，向对方要求证书和相关信息。
 - @收到含有Certificate Request载荷的ISAKMP报文的接收方，要用Certificate载荷（类型编号6）发送它的证书。

21

IKE的体系结构

23

IKE的两个阶段

第一阶段：
协商IKE SA

主模式

快速模式

第二阶段：
协商IPsec SA

快速模式

25

IKE的第一阶段：协商IKE SA

@IKE在第一阶段中要协商建立IKE SA，建立一个经过验证的安全通道，为后续的协商提供机密性和完整性保护。

@必须协商的内容包括：加密算法、哈希算法、认证(验证)方法、进行DH操作所使用组的有关信息等。

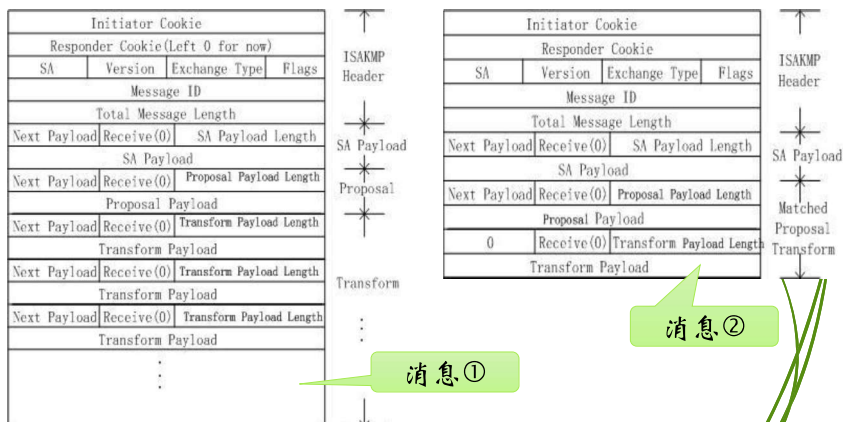
@IKE第一阶段中可以使用主模式和积极模式，而且这两种模式只能在第一阶段中使用。

@主模式提供了对通信双方的身份保护。

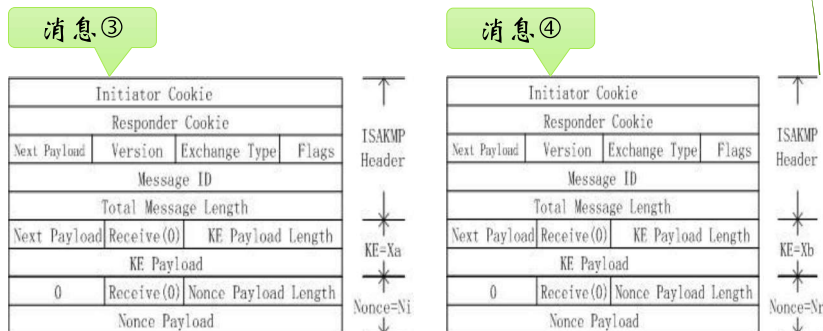
@当身份保护不必要时，可以使用积极模式以减少信息传输的数量，提高协商效率。

26

SA交换：消息①和②

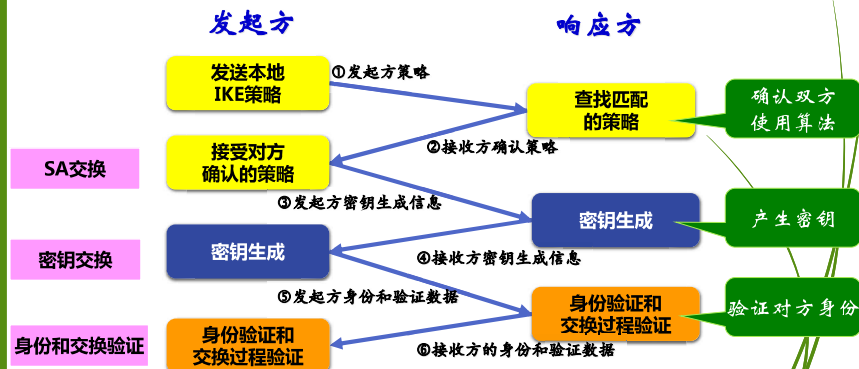


密钥交换：消息③和④



30

IKE的第一阶段：主模式



27

SA交换：消息①和②

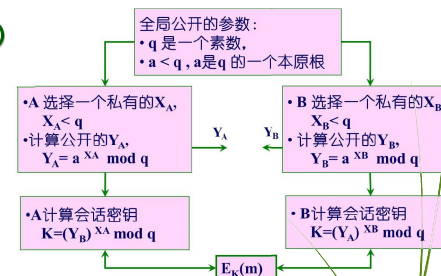
@在协商之前，发起者和响应者必须计算产生cookie，用于唯一的标识每个单独的协商交换过程。

@cookie使用源/目的IP地址、随机数字、日期和时间等信息进行MD5计算，其结果写入ISAKMP报文头的“cookie”域中。

@发起者发送消息①，接受者响应消息②；双方就散列函数、加密算法、认证方法、IKE SA协商的时间限制等进行了协商。

密钥交换：消息③和④

@发起方和响应方采用Diffie-Hellman密钥交换算法计算共享密钥。



@发起方发送消息③，响应方发送消息④：

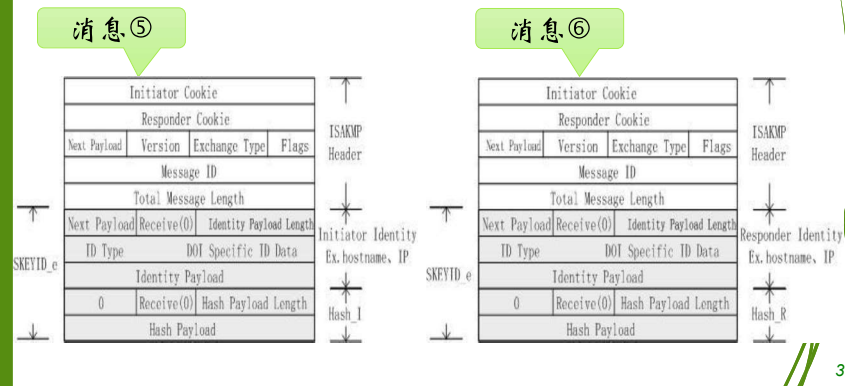
@消息③④的ISAKMP报文头和消息①和②相同，但是载荷不同

@消息③包含了Key Exchange载荷和Nonce载荷

@消息类型4和消息类型10

31

身份和交换验证：消息⑤和⑥



IKE的两个阶段

第一阶段：
协商IKE SA

主模式

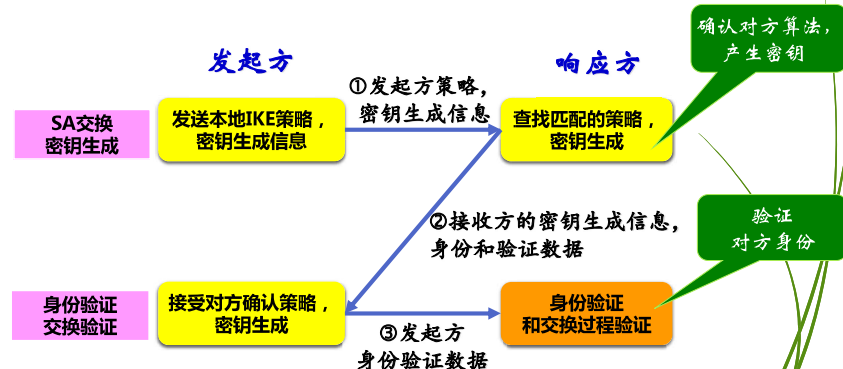
快速模式

第二阶段：
协商IPsec SA

快速模式

32

IKE的第一阶段：快速模式



身份和交换验证：消息⑤和⑥

①发起方发送消息⑤，ISAKMP头还是和前面的一样，但是载荷不同，消息⑤中的载荷包含标识载荷和散列载荷，并且加密传输。

②身份认证载荷（消息类型5）包含了发起者的标识信息，IP地址或者主机名；散列载荷（消息类型8）包含对上一过程中产生的三组密钥进行Hash运算得出的值。

③这两个载荷加密传输。

④消息⑥是响应方发送的，包含了响应者相应的信息。

⑤如果双方在消息⑤和消息⑥中的散列载荷中的hash值相同，那么双方的认证成功。**IKE第一阶段协商顺利完成。**

33

IKE的第一阶段：快速模式

①IKE为什么需要快速模式：

②适用于一方地址为动态的情况

③主模式只能采用IP地址作为ID进行协商，不能够应用于IP地址变化的情况

④快速模式可以采用Name方式进行验证，可以应对IKE协商双方中有一方为动态地址的情况；但是，快速模式不能用于两端地址都变化的情况

⑤快速模式传输的消息更少，效率更高

⑥主模式需要交换6个消息，快速模式只需要交换3个消息

35

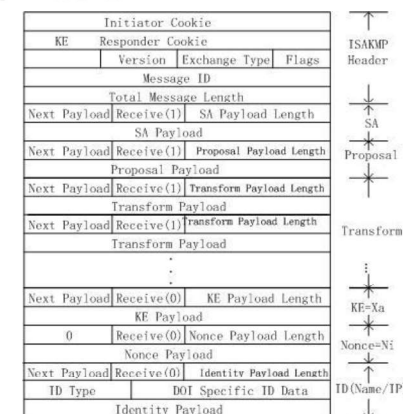
快速模式：消息①

①在协商之前，发起者发送前计算好cookie

②消息①交换SA、KEY、Nonce和ID等载荷

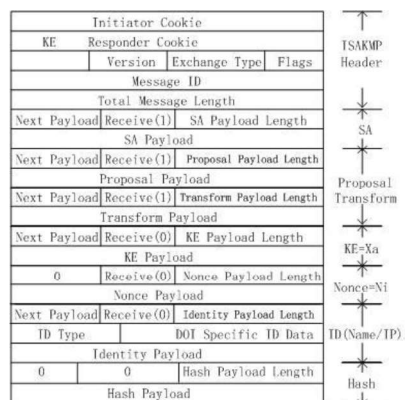
③消息②在响应消息①内容的同时，增加了Hash认证

④消息③是响应方对发起方的认证



快速模式：消息②

- 消息②在交换消息①内容的同时，增加了Hash认证
- 与主模式中的消息⑤⑥的准备工作一样，响应方计算好几个密钥，然后使用这些密钥以及其他信息计算hash值。
- 消息②回答了消息①的相应内容，还包括了计算好的hash值。



IKE的两个阶段

第一阶段：
协商IKE SA

主模式

快速模式

第二阶段：
协商IPsec SA

快速模式

40

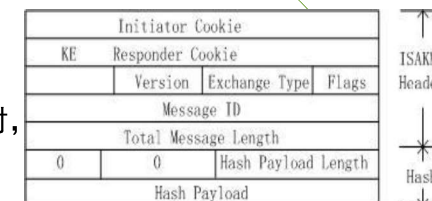
IKE的第二阶段：需要协商的参数

- 加密算法：包括DES、IDEA、Blowfish、3DES、CAST等
- 哈希算法：包括MD5、SHA、Tiger等
- 验证方法
 - 包括共享密钥、RSA签名、DSS签名、RSA加密、RSA加密等
- DH组
- 存活周期类型及长度：类型包括秒,千字节等
- 密钥长度
- 等等

42

快速模式：消息③

- 发送方根据消息②传来的密钥，计算Hash值；如果这个Hash值和消息②中的Hash值一致，则发送方验证响应方成功。然后，再计算一个新的Hash值，加载在消息③中，发送给响应方。
- 响应方收到了消息③后，计算自己的Hash值，与消息③的Hash值进行比对，如果一致，认证成功。
- 快速模式顺利完成。

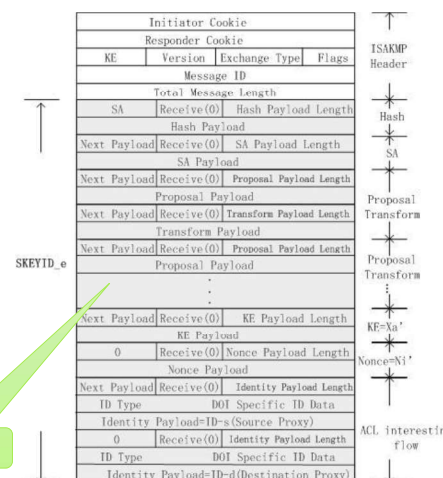
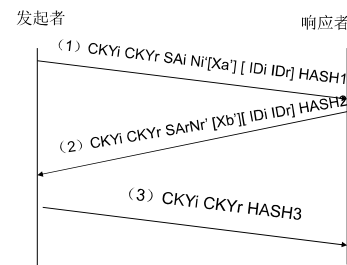


IKE的第二阶段：建立IPsec SA

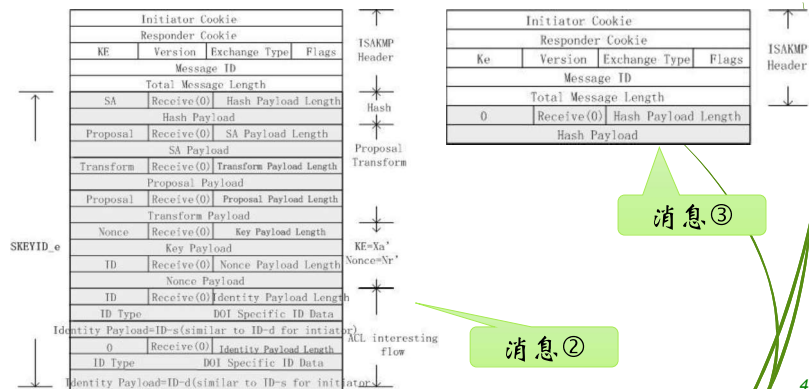
- 一个IKE SA协商（第一阶段）可为多个IPsec SA协商（第二阶段）提供服务。
- 第二阶段为IPsec AH 和 / 或 IPsec ESP协商安全参数，还可以实现密钥数据的交换。
- 第二阶段协商的内容与所协商的安全协议等相关。
 - 例如IPsec AH 协议需要协商认证算法，IPsec ESP 协议需要协商认证和加密算法。
- 第二阶段中使用快速模式进行信息交换，一个第二阶段协商可以建立多个安全关联。
 - 例如用于两个通信实体的双方向的共8个安全关联：每个通信实体都包含一个出AH SA、一个入AH SA、一个出ESP SA、一个入ESP SA。

41

IKE的第二阶段：消息交互举例



IKE的第二阶段：消息交互举例

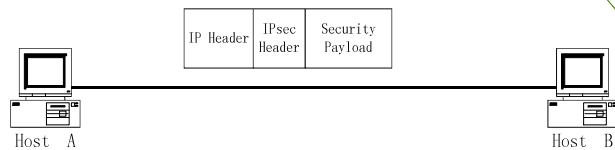


44

IKE的工作模式

@端节点到端节点传输模式

- @两个端节点均实现IPsec，通常使用传输模式。
- @IP头与数据间插入IPsec头，用来保护数据载荷。



46

IKE的工作模式

@其它模式

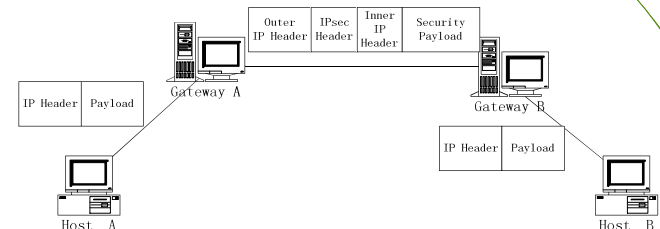
- @常见的上面两种模式的嵌套组合
- @例如：端节点到安全网关模式。
 - @假设一个便携漫游用户通过IPsec保护的隧道连接回自己的协作网络。在这种情况下，数据包将使用隧道模式。
 - @从便携漫游用户节点出来的数据包的外部IP头将包含和其当前位置相关的IP地址，而内部IP头将包含由安全网关赋予的源IP地址。
 - @外部目的地址总是安全网关的地址，而内部目的地址会是数据包的最终地址。

48

IKE的工作模式

@安全网关到安全网关隧道模式

- @端系统均不需要实现IPsec，由网络节点完成保护功能。通常采用隧道模式发送，内部IP头包含实际端节点的IP地址。

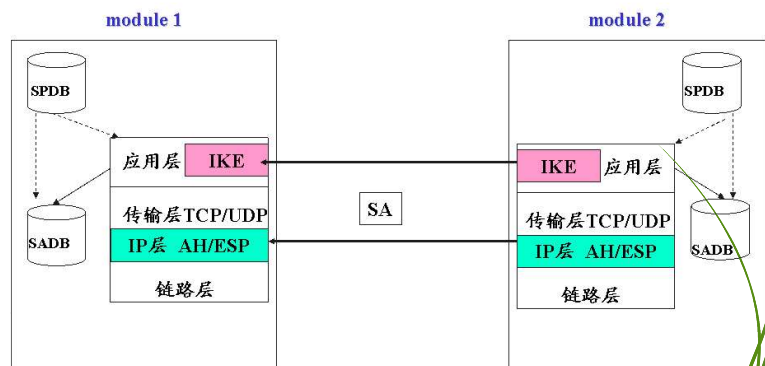


47

IKE的工作过程

49

IKE的工作过程



50

IKE的工作过程

@SPDB在每个条目中隐藏有指针。无论对于外出IP包还是进入IP包，首先都要对SPDB表进行查询，以决定是否丢弃、绕过或应用IPsec。

②对于需要处理的外出包，IPsec查询SADB，检查是否拥有合适的SA：如果有，则进行相应的IPsec处理；如果没有，就会向IKE守护进程发出创建SA请求。

@IKE守护进程接收到内核发来的请求后，查询SPDB，得到所有协商参数。然后向远程IKE进程发出协商请求；IKE进程开始协商。

52

IKE协议总结

54

IKE的工作过程

@IKE在协议实现上，是以守护进程的方式在后台运行。

@可以通过两种方式来启动IKE服务，由内核提交创建IKE SA请求，或同级IKE守护进程提交协商SA请求。

@两个守护进程通过UDP协议（端口500）来传递消息。

@IKE协议使用两个数据库：安全关联数据库SADB和安全策略数据库SPDB，这两个数据库都保存在操作系统内核。

51

IKE工作过程



@如果协商成功，把新协商的SA增加到SADB中。如果因为SPDB参数问题，协商未成功，IKE进程给策略及SA管理模块提示，由管理员来配置SPDB参数。

@当管理员指示IKE守护进程不再使用某个SA时，IKE守护进程会从SADB中删除掉相应的记录SA，同时会向远地的IKE守护进程发送删除信息，表示本地已经不再使用此SA了。

@远地IKE进程收到此信息后，根据实际的要求作不同的处理：

②可以选择删除相应的SA。

②也可以选择忽略此信息，继续保留相应的SA，但是不允许使用此SA继续通信。

53

IKE协议的不足

@标准定义的复杂性（需要参考3个RFC）容易导致理解上的困难和实现上的混乱，最终会导致不同实现间的互操作困难。

@完成协商的消息往返次数过多，这样会消耗较多的计算以及网络带宽资源。

@容易受到各种攻击，由于设计缺陷，协议实现容易受到拒绝服务攻击、中间人攻击、重放攻击等多种攻击行为的威胁。

55

IKE协议的总结



@IKE是一组复杂混合协议集合，已经发展成为Internet标准

@主要用途：在IPSEC通信双方之间建立共享安全关联SA

@一些困难

@太复杂

@协议复杂性，导致很难达到真正的安全性

@管理和配置复杂，使得安全性下降

@易受攻击

@还在进一步完善---IKE2

课程主要内容

- @一、计算机网络安全概论
- @二、密码学基础
 - @密码学的基础知识
 - @对称密钥密码算法
 - @公开密钥密码算法
 - @密钥的分配
- @三、认证技术
 - @消息认证技术
 - @身份认证及其实例
- @四、网络安全协议
 - @网络层安全：IPsec、IKE
 - @**传输层安全：SSL**
 - @应用层安全：电子邮件安全、电子商务安全、WEB安全
- @五、各类专用技术

2

传输层安全协议SSL概述

4

传输层安全协议：SSL和TLS

- @1994年，Netscape Communication公司设计开发安全套接字协议SSL(Secure Socket Layer)，专门用于保护Web通讯。
 - @SSLv1.0：不成熟
 - @SSLv2.0：基本上解决了Web通讯的安全问题
 - @SSLv3.0：
 - @1996年增加了一些算法，修改了一些缺陷
 - @1997年，SSLv3.0作为RFC发布，IETF建立一个TLS (Transport Layer Security) 工作小组，作为互联网标准。
 - @1999年，TLSv1.0发布 (RFC 2246)，可以看作是SSLv3.1。

6

传输层安全协议SSL

SSL协议概述

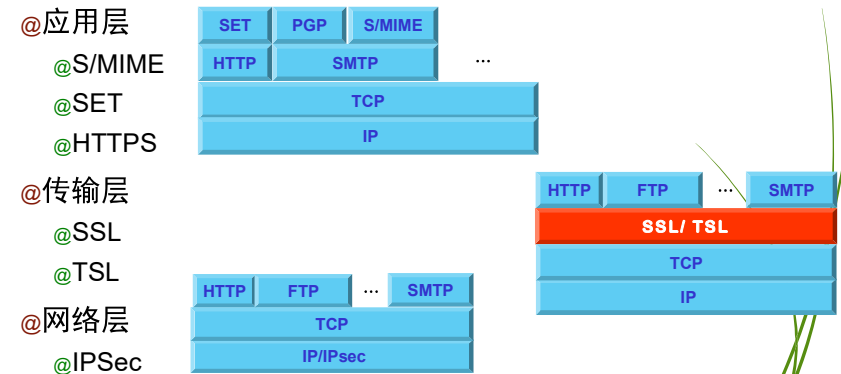
SSL体系结构

- SSL记录协议
- SSL握手协议
- SSL告警协议
- SSL修改规格协议

SSL安全性分析

3

安全通信协议



5

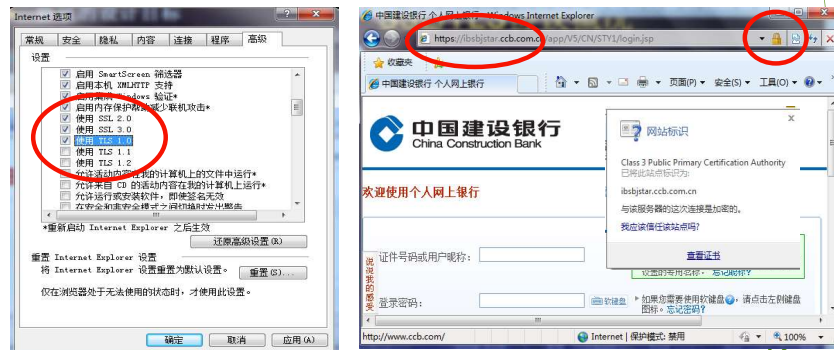
SSL的设计目标

- @SSL工作在TCP协议上，目前不支持UDP。
- @SSL与应用层协议无关，可以支持HTTP、Telnet、FTP等面向连接的应用层协议。
- @SSL为端到端的应用提供**保密性、完整性、身份认证**等安全服务，具有良好的互操作性、可扩展性和相对高的效率。
 - @SSL使用了加密技术、证书认证、数字签名等多种技术，可以提供数据加密、报文完整性检查、服务器认证、可选客户认证等。
- @采用SSL协议，可确保信息在传输过程中不被修改，目前在互联网上广泛用于处理财务等敏感信息。

7

SSL的使用案例

@凡是支持SSL协议的网页，都以https://作URL的开头



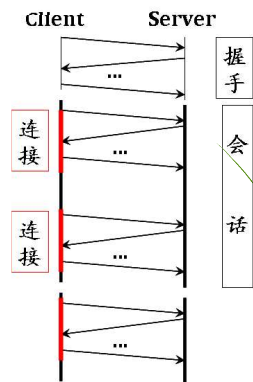
SSL体系结构

SSL的会话与连接

@连接（Connection）

@连接是一个特定的通信信道，通常映射成一个TCP连接。

@连接一般是短暂的，比如HTTPS中的一次访问中可能需要多个连接，共享同一个会话协商的密码算法、主密钥等信息。



SSL的使用案例

@如果利用SSL来访问网页，其步骤如下：

@用户：浏览器里输入 https://www.sslserver.com

@HTTP层：将用户需求翻译成HTTP请求

@SSL层：借助下层协议的信道安全的协商出一份加密密钥，并用此密钥来加密HTTP请求。

@TCP层：与web server的443端口建立连接，传递SSL处理后的数据。接收端与此过程相反。

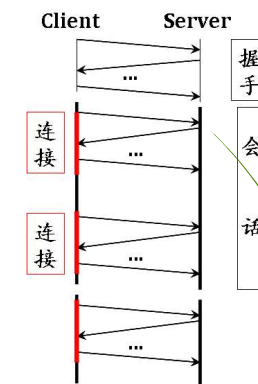
@SSL在TCP之上建立了一个加密通道，通过这一层的数据经过了加密，因此达到保密的效果。

SSL的会话与连接

@会话（Session）

@SSL会话是客户端和服务端之间的一个关联(association)，即一个虚拟的连接关系。

@通过握手协议建立，用以协商密码算法、主密钥等信息，一个会话协商的信息可以由多个连接共享。



会话状态

@会话状态用以下几个参数描述

@会话标识（Session ID）

@服务器用于标识会话活动或恢复状态所选的一个随机字节序列。

@同位体证书（Peer certificate）

@同位体的 X.509v3证书，此状态元素可以为空。

@压缩方法（Compression method）

@在加密前使用的压缩数据算法

@密码规范（Cipher specification）

@主要数据的加密算法(e.g. DES), 和计算MAC的算法 (e.g. MD5, SHA-1), 同时也定义相关属性。

@主密码（Master secret）

@客户和服务端之间48字节的共享密码。

@可恢复性（Is resumable）

@表明会话是否可被用于初始化新连接的标志。

连接状态

@连接状态用以下几个参数描述

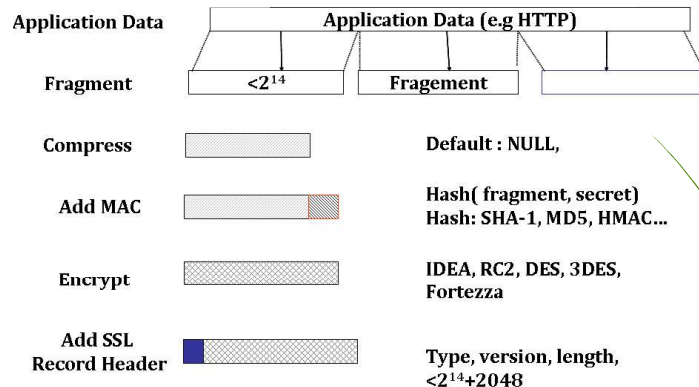
- @服务器和客户端随机数 (Server and client random)
 - @服务器和客户端为每个连接随机选择的字节序列。
- @服务器写MAC密码 (Server write MAC secret)
 - @服务器生成MAC时使用的密钥。
- @客户端写MAC密码 (Client write MAC secret)
 - @客户端生成 MAC时使用的密钥。
- @服务器写密钥 (Server write key- Client read key)
 - @服务器加密和客户端解密时使用的传统加密密钥
- @客户端写密钥 (Client write key- Server read key)
 - @客户端加密和服务器解密时使用的传统加密密钥
- @初始化矢量IV (initialization vectors)
 - @每个密钥需要维护一个初始化矢量。
- @序列号 (Sequence numbers)
 - @会话的各方为每个连接传送和结构消息维护的一个单独的序列号。

14

SSL记录协议

16

SSL记录协议的操作过程



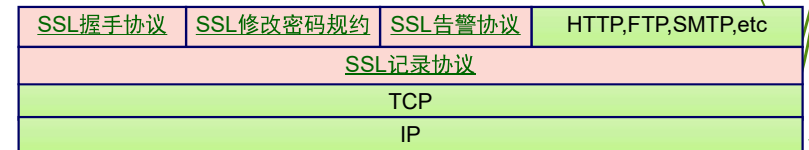
18

SSL体系结构



@SSL协议由以下协议组成

- @SSL记录协议 (SSL Record Protocol)
 - @定义传输格式，为高层协议提供数据封装、压缩、加密等基本功能
- @SSL握手协议 (SSL Handshake Protocol)
 - @协商密钥，在数据传输开始前，进行身份认证、协商加密算法、交换密钥等。
- @SSL告警协议
- @SSL修改密码规约协议



15

SSL记录协议

@SSL记录协议为SSL连接提供两种服务。

@保密性(报文的加密、解密):

使用SSL握手协议定义的共享密钥 (shared secret Key) ,
用传统密钥算法对SSL载荷进行加密。

@报文完整性:

用握手协议定义的共享密钥 (secret) 计算报文认证码
(MAC)。

17

SSL记录协议的操作过程

@分段

- @每个上层报文被分成 <2¹⁴ 字节(16384字节)的数据块或更小。

@压缩 (可选)

- @压缩不能丢失信息，并且对于短的数据块，压缩后增加内容不能超过1024字节。

@增加MAC

- @在压缩数据基础上计算MAC。需要使用共享密钥。

@加密

- @使用对称加密算法对压缩报文加上MAC进行加密。

@增加首部

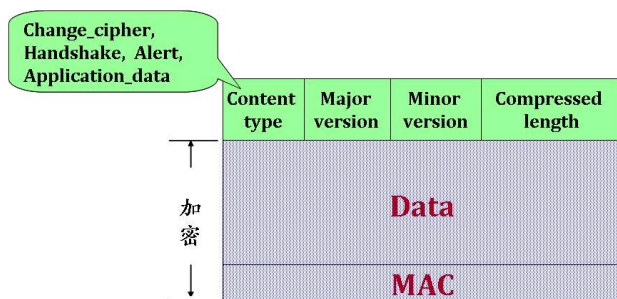
- @内容类型(8bit)，主要版本(8bit)，次要版本(8bit)，压缩长度(16bit)。

19

SSL记录协议的操作过程



@最后，形成SSL 记录协议数据单元。



20

SSL握手协议

@SSL的部分复杂性来自于握手协议，握手协议在传递应用数据之前使用。

@SSL握手协议允许客户端和服务端相互认证、协商加密和MAC算法，保护数据使用密钥通过SSL记录传送。

@SSL握手协议的功能包括：

@协商密码算法

@协商主会话密钥（Master Secret）

@Client 和Server之间的相互认证：

@对服务器身份的认证要在Client身份认证之前

@Client身份认证是可选的

22

握手协议消息类型

报文类型 (type)	参数 (Content)
hello_request	空
client_hello	版本,随机数,会话ID,密文族,压缩方法
server_hello	版本,随机数,会话ID,密文族,压缩方法
certificate	X.509v3证书链
server_key_exchange	参数、签名
certificate_request	类型、CA
server_done	空
certificate_verify	签名
client_key_exchange	参数、签名
finished	散列值

24

SSL握手协议

@握手协议由一系列在客户端和服务端之间交换的报文组成。认证、密钥和算法协商

@每个报具有三个字段：

@类型(1字节)：指示10种报文中的一个

@长度(3字节)：以字节为单位的报文长度

@内容(>=1字节)：和这个报文有关的参数

1 byte	3 bytes	0 bytes
Type	Length	Content

23

SSL 握手协议

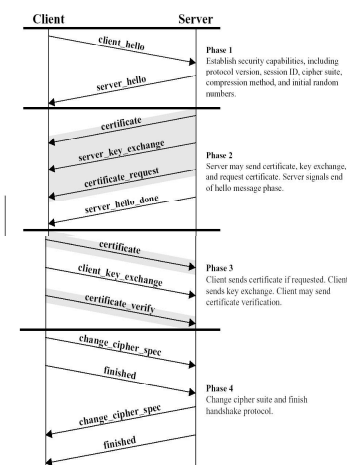
@四个阶段

@Phase 1. Establish security Capabilities

@Phase 2. Server Authentication and Key Exchange

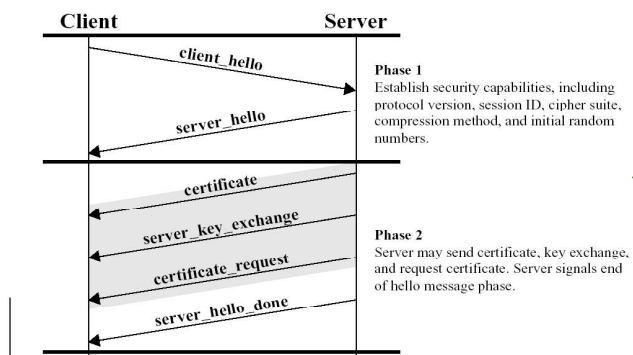
@Phase 3. Client Authentication and Key Exchange

@Phase 4. Finish



25

Phase 1, 2



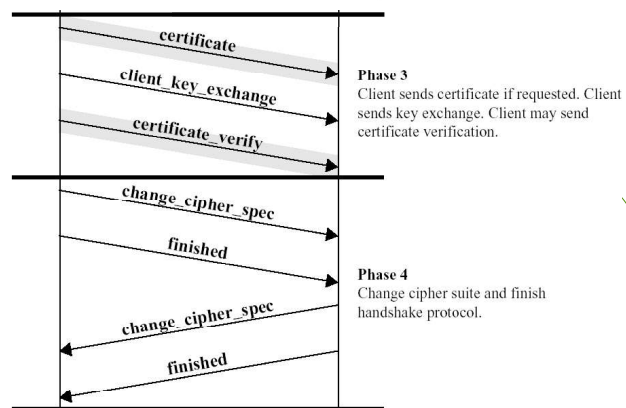
26

Phase 2 : 服务器认证和密钥交换

- @服务器发送其SSL数字证书Certificate（服务器使用带有SSL的X.509 V3数字证书）
- @（可选）如果服务器使用SSL 3.0，服务器的web应用程序需要客户端提交数字证书进行认证，则发出certificate_request报文。在报文中，服务器列出支持的客户端数字证书类型的列表和可接受的CA的名称。
- @SSL双向认证

28

Phase 3, 4



30

Phase 1 : 建立安全能力

- @客户向服务器发送client_hello报文
 - @参数包括：Version, Random, session id, cipher suite, compression method
- @服务器向客户端发送server_hello报文
 - @参数与client_hello参数相同
- @Cipher suite 参数包括
 - @密钥交换方法：RSA, Diffie-Hellman, Fortezza
 - @加密算法：RC4, RC2, DES, 3DES, IDEA, Fortezza
 - @MAC算法：MD5, SHA-1

27

Phase 2 : 服务器认证和密钥交换

- @如果密钥交换算法使用匿名DH，就不需要服务器发送证书。难防止中间人类型的攻击。
- @如果密钥交换算法使用匿名DH、Ephemeral DH、RSA、Fortezza等，需要发送server_key_exchange报文，用来交换密钥
- @服务器发送server_hello_done报文，等待客户端响应

29

Phase 3 : 客户认证和密钥交换

- @收到服务器发来的server_hello_done以后，客户端（Web浏览器）验证server提供的证书是否有效。
- @（可选）如果客户端收到了certificate-request报文，则发送一个SSL数字证书Certificate，没有合适的数字证书是可用的，客户端将发送no_certificate alert，如果客户端数字证书认证是强制性的话，服务器应用程序将会使会话失败。

31

Phase 3 : 客户认证和密钥交换

@客户端发送client_key_exchange, 此消息包含 pre-master secret (46 字节的随机数字), 和 消息认证代码 (MAC) 密钥 (用服务器的公用密钥加密的)。

@在以后的阶段, pre_master_secret 用来计算master_secret.

@如果客户端发送了Certificate给服务器, 客户端将发出签有客户端专用密钥的certificate_verify, 通过验证此消息的签名, 服务器可以显示验证客户端数字证书的所有权。

32

Phase 4 : 结束



@客户端使用一系列加密运算将 pre-master secret 转化为 master secret, 其中将派生出所有用于加密和消息认证的密钥。

@客户端发出change_cipher_spec, 将服务器转换为新协商的密码对。

@客户发送在新算法、密钥、密码之下的finished, 验证密钥交换和认证过程是否成功。

@服务器发送change_cipher_spec, 将挂起状态迁移到当前的 Cipher Spec, 并发送结束报文。

@握手完成, 客户和服务器可以交换应用层数据。

34

SSL告警协议

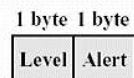


@告警协议用于向对等实体传递SSL相关的警报。和使用SSL的其它应用一样, 告警消息按照当前状态压缩和加密。

@此协议的每个消息由两个字节组成。

@第一个字节, 值为1标识告警, 值2表示致命错误来传递消息出错的严重程度。如果结果为致命, SSL将立即中止连接, 而会话中的其它连接将继续进行, 但不会在此会话中建立新连接。

@第二个字节, 包含了描述特定告警信息的代码。



(b) Alert Protocol

36

认证密钥和加密密钥

@主密钥master_secret并不直接用于数据加密和认证, 而是用来产生连接所需要的一系列密钥, 包括:

	Client	Server
MAC	client_write_secret	server_write_secret
加密	client_write_key	server_write_key
IV	client_write_iv	server_write_iv

33

SSL告警协议

35

SSL修改密码规约协议

37

SSL修改密码规约协议

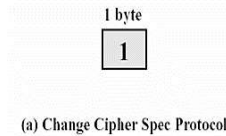


@功能

- @在握手协议的结束阶段发送，通知接收方，以后的记录将使用刚才协商的密码算法和密钥进行加密/认证。
- @接收方把会话的挂起状态复制到当前状态，使以后的连接使用这些密码参数。

@报文结构

- @只有一种报文，只有一个字节，只有一个值(1)是有效的。



38

SSL的安全性分析

- @几乎所有操作平台上的WEB浏览器以及流行的Web服务器都支持SSL协议，因此使用SSL协议开发成本小。
- @但是，SSL协议的数据安全性其实就是建立在RSA等算法的安全性上，从本质上来讲，攻破RSA等算法就等同于攻破此协议。
- @由于美国政府的出口限制，使得进入我国的实现了SSL的产品（Web浏览器和服务端）均只能提供512比特RSA公钥、40比特对称密钥的加密。系统安全性差。

40

SSL安全性分析



- @一个安全协议除了基于其所采用的加密算法安全性以外，更为关键的是其逻辑严密性、完整性、正确性。从目前的应用情况看来，SSL比较好地解决了这一问题。
- @另外，SSL协议可以很好地防范“重放攻击”。
 - @SSL协议为每一次安全连接产生了一个128位长的随机数——“连接序号”。理论上，攻击者提前无法预测此连接序号，因此不能对服务器的请求做出正确的应答。
- @随着SSL协议的不断改进，更多的安全性能、好的加密算法被采用，逻辑上的缺陷被弥补，SSL协议的安全性能会不断加强。

41

课程主要内容

- @一、计算机网络安全概论
- @二、密码学基础
 - @密码学的基础知识
 - @对称密钥密码算法
 - @公开密钥密码算法
 - @密钥的分配
- @三、认证技术
 - @消息认证技术
 - @身份认证及其实例
- @四、网络安全协议
 - @网络层安全：IPsec、IKE
 - @传输层安全：SSL
 - @应用层安全：电子邮件安全、电子商务安全、WEB安全
- @五、各类专用技术
 - @访问控制技术：原理、实践

2

电子邮件系统安全需求

4

电子邮件系统安全问题

- @垃圾邮件和邮件炸弹
 - @垃圾邮件是未经用户请求，强行发送到用户信息中的电子邮件。
 - @邮件炸弹是指发送大量的垃圾邮件，从而塞满邮箱，大量占用了系统的可用空间和资源，使得用户终端暂时无法正常工作。
- @邮件病毒
 - @通过电子邮件进行计算机病毒传播，通常以附件形式，传播带有病毒、蠕虫、木马程序等的电子邮件，将对接收方的计算机系统和整个计算机网络带来严重破坏。

安全的电子邮件

- @电子邮件系统安全需求
- @电子邮件系统工作原理
 - @RFC 822、SMTP、POP3、IMAP4
 - @MIME
- @安全电子邮件协议标准
 - @PEM、MOSS、S/MIME、PGP
- @S/MIME
- @PGP
- @反垃圾邮件常用技术



3

电子邮件系统安全问题

- @匿名转发
 - @发件人刻意隐瞒自己的电子邮箱地址和其他信息，或者通过某些方法一些错误的发件人信息
 - @发送者首先将邮件发送给匿名转发系统，匿名转发邮件系统再把邮件转发给真正的收件者，并将匿名转发邮件系统的地址作为发信人地址显示在邮件的信息表头中。
- @电子邮件欺骗
 - @假冒某一个用户的身份给其他用户发送邮件。SMTP本身并不提供任何验证，伪造者只要给SMTP服务器提供合适的信封信息，并使用想要的的数据产生有关的信件即可。
 - @与假冒相关的另一个安全问题是抵赖问题，因为信件可以伪造，发送方也可以否认曾经发送过某个信件。

如何解决上述问题

- @端到端的安全电子邮件技术，保证邮件从被发出到被接收的整个过程中，内容保密、无法修改、并且不可否认。
 - @目前常用的安全电子邮件标准是S/MIME和PGP，它们一般只对信体进行加密和签名，而信头必须保证原封不动。
- @在一些应用环境下，可能会要求信头在传输过程中也保密，这就需要借助传输层安全技术。
 - @方法一：利用SSL SMTP和SSL POP。
 - @要求客户端的电子邮件软件和服务器端的邮件服务器都支持SSL，都必须安装SSL证书。
 - @方法二：利用VPN，将所有的TCP/IP传输封装起来，也包括邮件。

如何解决上述问题



@邮件服务器本身还必须安全可靠，目前对邮件服务器的攻击主要分为：

@防范网络入侵主要依赖于软件编程时的严谨程度，很难从外部衡量。

@对于拒绝服务攻击的防范，主要包括：

@防止来自外部网络的攻击

@防止来自内部网络的攻击

@防止中继攻击等

@总之，电子邮件系统的安全需求包括：

机密性、完整性、可认证、抗否认

10

电子邮件简介

@背景介绍

@1972年，email出现

@1982年，SMTP协议出现

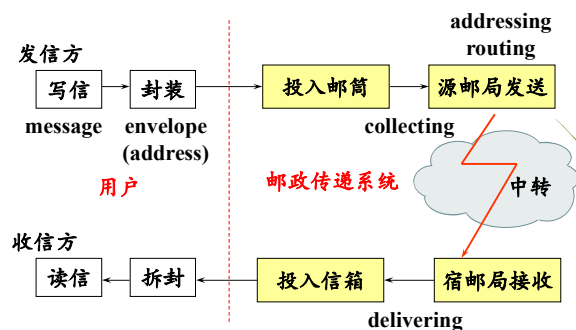
@2002年，CNNIC调查最常用服务email，92.2%

@2010年，微信等新型应用层出不穷，email依旧具有不可替代性

@电子邮件系统

@MTA、MUA、MDA等

传统邮政系统的构成



12

电子邮件系统工作原理

9

电子邮件概述

@电子邮件 Electronic Mail (E-mail)

@包含了实现用户之间非实时通信的应用程序和发送、接收、中继（中转）电子邮件的处理程序。需要解决的问题是寻址和路由。

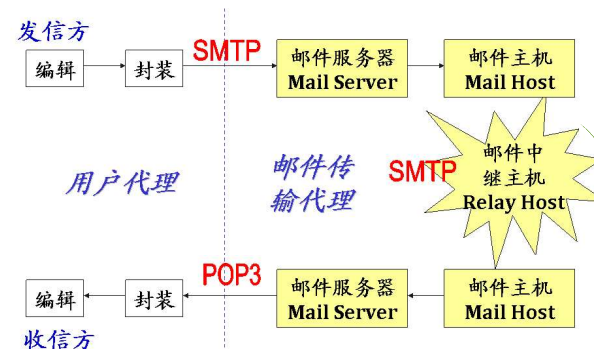
@每个用户首先应拥有一个电子邮箱。所谓的电子邮箱就是在台提供了邮件服务的主机上为每个人提供一个用户名。

@电子邮件的地址表示为：用户名@主机名。地址的大小写不与区分。

@ kitty@263.net

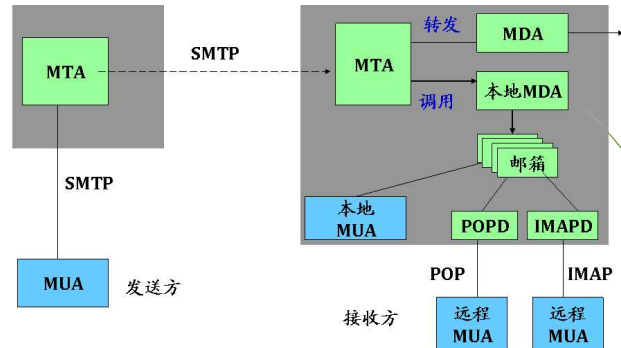
11

电子邮件系统的构成



13

电子邮件系统的工作原理



14

电子邮件主要相关协议

- @SMTP
 - @Simple Mail Transfer Protocol, 简单邮件传输协议 (RFC821)
- @POP3
 - @Post Office Protocol 3, 邮局协议版本3 (RFC1939)
- @IMAP4
 - @Internet Message Access Protocol 4, Internet消息访问协议版本4 (RFC2060)
- @RFC822
 - @Format of Electronic Mail Messages, 电子邮件消息格式
- @MIME
 - @Multipurpose Internet Mail Extensions, 多用途邮件扩展协议 (RFC 2045)

16

RFC 822、SMTP、POP3、IMAP4

18

Keywords

- @MTA----Mail Transfer Agent
- @MDA----Mail Delivery Agent
- @MUA----Mail User Agent
- @SMTP----Simple Mail Transfer Protocol
- @POP----Post Office Protocol
- @IMAP-----Interactive Mail Access Protocol

15

电子邮件主要相关协议

- @服务器之间的通信
 - @MUA $\leftrightarrow$ MTA, MTA $\leftrightarrow$ MTA,
 - @SMTP、ESMTP 等共25篇RFC
- @客户-服务器
 - @MUA $\leftrightarrow$ MDA
 - @POP3、IMAP等共30篇RFC
- @编码格式
 - @RFC 2822 等5篇
- @MIME: 95篇RFC
- @网关接口: 40多篇
- @国际化: 20多篇

17

RFC 822

- @RFC 822定义了一种用于电子邮件传输的文本消息格式，是基于互联网传递的文本邮件消息标准，并被广泛使用。
- @RFC 822中，消息包括信封和内容两个部分，信封包含了用于传递和发送的任何信息，内容是要发给接收方的对象。
- @一个消息包括若干行的报头和无限制的正文。
 - @报头和正文之间用一个空行隔开，消息是ASCII码文本。
 - @报头通常用关键字、冒号和该关键字的参数构成，并允许长行被分割为若干个短行。
 - @常用关键字有From/To/Subject/Date等。
 - @消息标识域Message-ID是与消息有关的唯一标识。

19

RFC822

```
Date: Tue, 16 Jan 1998 10:37:17 (EST)
From: "William Stallings" <ws@shore.net>
To: dcs@mail.tsinghua.edu.cn
Subject: Examples of RFC 822
```

Message
Header

```
Hello, would you kindly make an effort
to insure that your explanations are in
English and not in that other language
you occasionally drift into?  Many
Thanks.
```

Message
Body

-W.S.

20

SMTP协议部分指令

- @HELO/EHLO : 标识自己的主机名
- @MAIL : 标识邮件的发送者
- @RCPT : 标识邮件的接收者
- @DATA : 发送邮件数据
- @REST : 中止当前处理并复位
- @VRFY : 验证地址是否存在
- @EXPN : 扩充邮件表
- @NOOP : 只返回OK响应码(200)
- @QUIT : 退出

22

RFC821

Example of the SMTP Procedure

This SMTP example shows mail sent by Smith at host Alpha.ARPA, to Jones, Green, and Brown at host Beta.ARPA. Here we assume that host Alpha contacts host Beta directly.

```
S: MAIL FROM:<Smith@Alpha.ARPA>
R: 250 OK
```

```
S: RCPT TO:<Jones@Beta.ARPA>
R: 250 OK
```

```
S: RCPT TO:<Green@Beta.ARPA>
R: 550 No such user here
```

```
S: RCPT TO:<Brown@Beta.ARPA>
R: 250 OK
```

```
S: DATA
R: 354 Start mail input; end with <CRLF>.<CRLF>
S: Blah blah blah...
S: ...etc. etc. etc.
S: <CRLF>.<CRLF>
R: 250 OK
```

The mail has now been accepted for Jones and Brown. Green did not have a mailbox at host Beta.

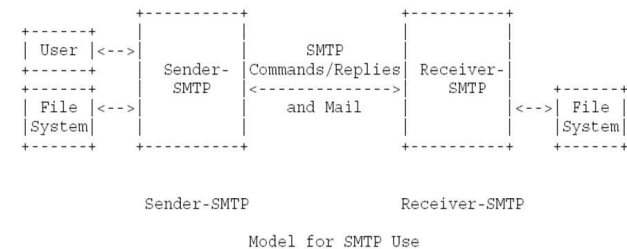
24

SMTP 协议

- @RFC 821, 1982年公布, TCP端口25。
- @常用命令
 - @HELO,MAIL,RCPT,DATA,RSET,VRFY,QUIT...
- @RFC822只适用于无格式的英文文本信息传输, 不能支持非ASCII码字符集信息, 不能处理中文、日文、包含格式的文本信息和多媒体信息。
- @SMTP没有提供任何安全服务。

21

RFC 821



23

利用SMTP发送邮件举例

```
$ telnet smtp.ccert.edu.cn 25
Trying 202.112.50.210...
Connected to smtp.ccert.edu.cn.
Escape character is '^]'.
220 ns2.ccert.edu.cn ESMTP Sendmail 8.12.1/8.12.1;
    Fri, 9 Aug 2002 22:19:18 +0800 (CST)
helo test.domain
250 ns2.ccert.edu.cn Hello incident.ccert.edu.cn
    [202.112.50.168],
    pleased to meet you
mail from:<god@heaven.net>
250 2.1.0 <god@heaven.net>... Sender ok
rcpt to:<duan@ccert.edu.cn>
250 2.1.5 <duan@ccert.edu.cn>... Recipient ok
```

25

利用SMTP发送邮件举例

```
data
354 Enter mail, end with "." on a line by itself
From: GOD<god@yahoo.com>
To: anybody<anybody@edu.cn>
Subject: Can you guess who am I ?

Spam email test
bye
.
250 2.0.0 g79EJIX2009423 Message accepted for
  delivery
quit
221 2.0.0 ns2.ccert.edu.cn closing connection
Connection closed by foreign host.
$
```

26

POP3协议

- @与SMTP协议相同，POP3也是个请求/响应协议。
- @其命令由短关键字构成，后面接着可选的参数，以CR和LF符结束，作为单行文本发送。例如：
 - @USER name —— 给出用户信箱名称
 - @PASS password —— 给出用户信箱口令
 - @STAT —— 请求服务器返回消息数和大小
 - @RETR msgnum —— 请求服务器发送指定的报文
 - @DELE msgnum —— 请求服务器删除指定的报文
 - @QUIT —— 结束会话

MIME

30

RFC 821/822 的主要限制

- @RFC 821(SMTP)不能传输二进制数据，如音频、视频等。
 - @虽然有一些方法可以将二进制文件转化为文本格式，如UNIX UUencode/ Uudecode模式，但是没有统一的标准。
- @SMTP不能传输非ASCII 码的字符
 - @ASCII码是用7bits描述的，不能描述8bits数据，如双字节语言等。
- @SMTP服务器可以拒绝超过一定大小的邮件信息。
- @SMTP网关在ASCII码与EBCDIC码之间转换时候没有使用统一的映射。

27

IMAP4协议

- @IMAP (Internet Message Access Protocol) 是一种强有力的邮箱访问方式，它为用户提供了有选择地从邮件服务器接收邮件的功能、基于服务器的信息处理功能和共享信箱功能。
- @与POP3的比较：
 - @POP3提供了快捷的邮件下载服务，用户可以把邮箱里的信下载到PC上进行离线阅读，并可以选择将邮件从服务器上删除。用户在任何时候都可阅读已经下载的邮件。
 - @IMAP同样提供了方便的邮件下载服务，让用户能进行离线阅读。另外，IMAP还提供的摘要浏览功能可以让用户在阅读完所有的邮件到达时间、主题、发件人、大小等信息后作出是否下载邮件的决定。配合IMAP客户端软件的支持，用户还可以有选择的下载附件。
 - @例如：一封邮件里含有5个附件，而其中只有2个附件是需要的，就可以只下载那2个附件，节省了下载其余3个的时间。

29

MIME : 概述

- @MIME(Multipurpose Internet Mail Extension)是对RFC 822框架的一个扩展。
 - @它期望以RFC 822兼容的方式解决现有电子邮件传输协议存在的一些问题和限制。
- @MIME中包含了如下要素：
 - @定义了五个新的报头域，这些域提供了正文信息。
 - @定义了若干内容转换，从而标准化地支持多媒体的电子邮件。
 - @定义了编码转换方式，使得任何内容格式均可转化为邮件系统认可的格式。

31

MIME：五个报头域

@MIME-version

@MIME版本域其值必须为1.0，说明该消息符合RFC 2045和2046。

@Content-type

@内容类型域描述了正文中包含的数据类型，使得接收方代理可以选择合适的代理或者机制表示数据或者正确处理数据。

@Content-Transfer-Encoding

@内容转换编码域描述了将消息正文转换为可以传输的类型转换方式。

@Content-ID

@内容标识域是在多重上下文中唯一标识MIME实体的标识。

@Content-Description

@内容描述域是正文对象的文本描述，在该对象不可读的时候使用，例如音频数据。

@RFC 822报头可以出现一个或者多个域，必须支持的是前三个域，可以选择支持后两个域，而且接收方可以忽略这两个域。

32

MIME：转换编码



@消息正文转换编码的目的是在庞大的网络环境中提供可靠的投递方式。

@MIME的Content-Transfer-Encoding域可取6个值。

Encoding	Description
7 bit	所有数据都用短行的 ASCII 码表示。
8 bit	短行，但可以有非 ASCII 码字符。
Binary	不仅可以有非 ASCII 码字符，而且可以存在 SMTP 无法传送的长行。
quoted-printable	如果被编码的数据大多是 ASCII 码字符，则数据编码后的形式大部分仍是可以识别的一种编码方式。
Base64	将 6 位输入转换为 8 为输出的方式，转换后的数据均为可以打印的 ASCII 码字符。
x-token	非标准编码方式的名称。

34

安全电子邮件标准1：PEM

@PEM: Privacy Enhanced Mail, RFC1421-1424

@PEM协议是80年代末90年代初，美国RSA实验室基于RSA和DES算法开发的早期标准。主要功能包括加密、源认证和完整性。

@PEM将邮件报文分成几部分，有的需要加密，有的则需要认证，这些需要特殊处理的报文通过语句标记。

@报文加密采用DES算法

@报文认证采用MD5算法

@密钥的分发：

@基于公钥技术，用接收者的公钥加密会话密钥，RFC1422中定义了证书体系。

@基于对称密码时，采用事先共享的密钥加密会话密钥。

@由于PME依赖于一个即存的、完全可操作的PKI，而建立PKI需要很长的过程，大大限制了PEM的发展，现在几乎已经没有人使用PEM了。

36

MIME：内容格式

@内容类型用于描述数据的通用类型，子类型描述该数据类型的特殊格式。

@RFC 2046中列出了7种类型和15种子类型。

Type	Subtype	Description
Text	Plain	无格式文本，为 ASCII 码或者 ISO 8859 码
	enriched	提供丰富的文档消息
multipart	Mixed	个部分相互独立，但一起传送。接收方看到的形式与邮件消息中的形式相同。
	Parallel	与 Mixed 基本相同，只是传送时候个部分无序。
	alternative,	不同部分是同一消息的不同表现形式，按增序排列，接收方系统将按照最佳方式显示
	Digest	与 Mixed 基本相同，但每部分默认的类型/子类型为 message/RFC 822。
message	rfc822	封装消息的正文与 RFC 822 一致。
	partial	允许按照接收方透明的方式对大段邮件分段。
	external-body	包含一个对存在于别处的对象的指针
image	jpeg	JFIF 编码的 JPEG 图像
	gif	GIF 格式的图形
video	Mpeg	MPEG 格式
audio	Basic	单通道 8 位 ISDN u 律，8 kHz 采样率
application	postscript	Adobe PostScript
	octet-stream	一般的 8 位二进制数据

35

安全电子邮件协议标准

安全电子邮件标准2：MOSS

@MOSS: MIME Object Security Services, RFC1848

@MOSS针对PEM的不足做了一些改进，改变了PEM只支持文本信息的局限性，支持了MIME，将PEM和MIME两者的特性进行了结合。

@但是，MOSS有很多的执行选项，这可能导致两个不同的开发人员提出的两种MOSS邮件无法沟通。

@MOSS往往被认为是一种框架而不是一个规范，在实现时还要考虑许多实际的问题，因此也没有被业界广泛使用。

37

安全电子邮件标准3：S/MIME

@目前常用的端到端安全电子邮件标准：PGP和S/MIME。

@S/MIME侧重于作为商业和团体使用的工业标准

@PGP则倾向于为用户提供个人电子邮件的安全

@S/MIME：Secure/Multipurpose Internet Mail Extension，RFC 2630/2632/2633等

@S/MIME是在PEM的基础上建立起来的，使用RSA的PKS#7标准，与MIME一起来保护电子邮件信息。

@S/MIME已经成为产业界广泛认可的协议，如Microsoft公司、Netscape公司、Novell公司、Lotus公司等都支持该协议。

38

S/MIME

40

S/MIME：功能

@S/MIME提供了签名和加密消息的能力。

@封装数据

@由任何类型的加密内容和加密该内容所用的加密密钥组成，密钥可以是一个或者多个接收方对应的多个密钥。

@数字签名

@通过提取待签名内容的数字摘要，用签名者的私钥加密。然后，用base64编码方法重新对内容和签名编码。因此，一个签了名的消息只能被具有S/MIME能力的接收方处理。

@轻载的透明签名：只有数字签名采用base64编码方法，没有S/MIME能力的接收方虽然无法验证签名，却可以看到消息内容。

42

安全电子邮件标准4：PGP



@PGP：Pretty Good Privacy

@PGP既是一个特定的安全电子邮件应用，也是一个安全电子邮件标准。

@PGP符合PEM的绝大多数规范，但不要求PKI的存在。

@PGP创造性地把RSA公钥体系的方便和传统加密体系的高度结合起来，并在数字签名和密钥认证管理机制上有非常巧妙的设计。

@PGP功能强大，速度快，源代码公开。

39

S/MIME：概述

@S/MIME（Secure/Multipurpose Internet Mail Extension）是从PEM和MIME发展而来的，提供统一的方法来接收和发送MIME数据，提供的安全服务包括认证、数据机密性、消息完整性和非否认。

@S/MIME目标是实现安全的电子邮件，侧重于作为商业和团体使用的工业标准。

41

S/MIME：密码算法

@S/MIME组合三种公钥算法

@DSS是其推荐的数字签名算法

@Diffie-Hellman是其推荐的密钥交换算法

@3DES是其推荐的消息加密算法

@另外，RSA既可以用做签名，也可以加密会话密钥。

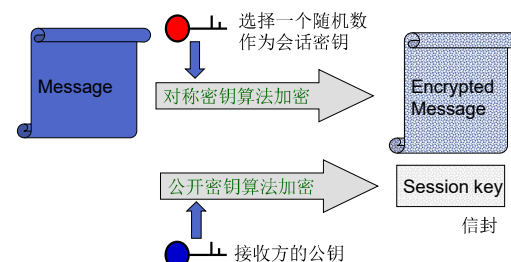
43

S/MIME : 密码算法

功能	要求
创建用于数字签名的数字摘要	必须支持 SHA-1, 接收方应该支持 MD5, 以便向后兼容
加密数字摘要形成数字签名	发送代理和接收代理必须支持 DSS 发送代理应该支持 RSA 加密 接收代理应该支持验证密钥大小在 512 位至 1024 位的 RSA 签名
为传送消息加密会话密钥	发送代理和接收代理必须支持 DH 密钥交换算法 发送代理应该支持验证密钥大小在 512 位至 1024 位的 RSA 加密 接收代理应该支持 RSA 解密
用一次性会话密钥加密消息	发送代理应该支持 3DES 和 RC2/40 加密 接收代理必须支持用 3DES 解密, 应该支持 RC2/40 解密

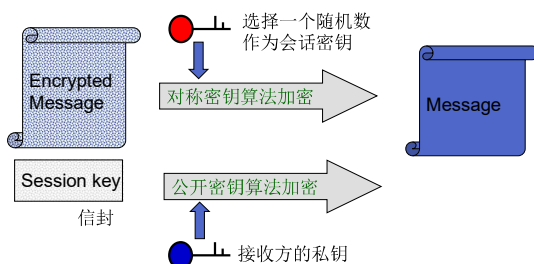
44

S/MIME : 加密过程



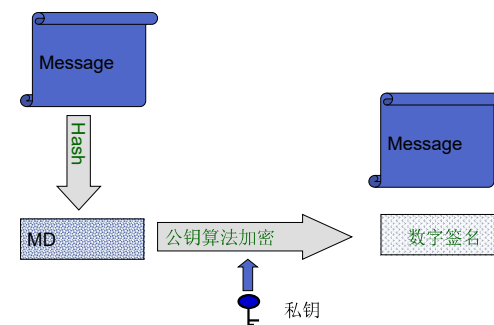
45

S/MIME : 解密过程



46

S/MIME : 数字签名过程



47

S/MIME : 消息处理过程

@S/MIME使用了一系列新的MIME内容类型, 所有的新类型都使用PKCS(Public-Key Cryptography Specifications)指示。

Type	SubType	S/MIME parameter	Description
Multipart	Signed	-	两部分的透明签名消息: 一部分是消息, 另一部分是签名。
application	Pkcs7-mime	signedData	签名的 S/MIME 实体
	Pkcs7-mime	envelopedData	加密的 S/MIME 实体
	Pkcs7-mime	Degenerate signedData	仅包含公钥证书的实体
	Pkcs7-signature	-	签名子部分的内容类型为 multipart/signed 的消息
	Pkcs10-mime	-	证书注册请求消息

48

S/MIME : 消息处理过程

@保护MIME实体

@S/MIME用签名和/或加密保护MIME实体。将MIME实体和一些与安全相关的数据(如算法标识符、证书)一起用S/MIME处理, 得到PKCS对象。最后, 将PKCS对象作为消息内容封装到MIME中。

@封装数据

@为特定的对称加密算法(RC2或3DES)生成伪随机的会话密钥。
 @用每个接收方的RSA公钥分别加密会话密钥。
 @为每个接收方准备一个接收方信息块, 包含接收方的公钥证书标识、加密会话密钥的算法标识和加密后的会话密钥。
 @用会话密钥加密消息内容。
 @接收方信息块后面跟着构成封装数据的加密内容, 然后用base64编码。为了恢复加密的消息, 接收方先去掉base64编码, 然后用其私钥回复会话密钥。最后用会话密钥解密得到的消息内容。

49

S/MIME : 消息处理过程

@签名数据

- @选择消息摘要算法(SHA或MD5)
- @计算待签名内容的消息摘要或者hash函数
- @用签名者的私钥加密数字摘要
- @准备一个签名者信息块, 包含签名者的公钥证书、消息摘要算法的标识符、加密消息摘要的算法标识符和加密的消息摘要。
- @签名数据实体包括一个消息摘要算法标识符、被签名消息和签名者信息块, 最后将这些数据用base64转换编码。
- @为了恢复签名消息并验证签名, 接收方首先去掉base64编码, 然后用签名者的公钥解密信息摘要, 接收方独立计算消息摘要, 并将其与解密得到的消息摘要进行比较, 从而验证签名。

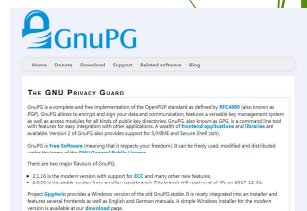
50

PGP

52

PGP发展历程

- @1996年, zimmermann创立了PGP公司, 其目的是使加密技术被普通人使用。
- @1997年, NAI公司收购了PGP公司, PGP成为商业版本, 不再提供免费版本。
- @2002年, PGP被一家名字为“PGP有限公司”的新公司购买。
 - @zimmermann一直没有离开, 他公开了PGP源码, 提供非商业性质的免费版本
 - @另外组建了openPGP联盟 (www.openpgp.org)
 - @其开源版本为gnupg, 下载地址www.gnupg.org



S/MIME : 消息处理过程



@透明签名

- @透明签名在对多部分内容类型的子类型签名时使用。签名过程并不涉及对签名消息的转换, 该消息发送的是明文。因此, 具有MIME能力而不具有S/MIME能力的接收方也能够阅读输入的消息。
- @一个multipart/signed的消息由两个部分构成。第一部分是任何MIME类型, 第二部分是数字签名, 但是其消息内容为空。
- @接收方从第一部分获得消息摘要, 并同第二部分中恢复得到的消息摘要进行比较来进行认证。

@注册请求

- @一个应用或者用户向认证中心申请公钥证书, S/MIME实体用application/PKCS10传递证书申请。

51

PGP

- @PGP (Pretty Good Privacy)由个人发展起来,为电子邮件和文件存储应用提供了认证和保密性服务

@Phil Zimmermann

@PGP特点

- @软件免费、版本众多、支持各种系统平台
- @采用一系列安全算法和机制, 安全性已经得到了充分的论证
 - @邮件和文件存储
- @自由发放, 包括文档、源代码等
 - @不是由政府或者标准化组织所控制



PGP提供的五种安全服务

@加密

- @发信人产生一次性会话密钥, 以IDEA、3-DES或CAST-128算法加密报文, 采用RSA或DH密钥交换算法用收信人的公钥加密会话密钥, 并和消息一起送出。

@认证

- @用SHA-1对报文哈希, 采用RSA或DSS以发信人的私钥签名。

@压缩

- @ZIP, 用于消息的传送或存储。在压缩前签字, 压缩后加密。

@兼容性

- @采用Radix-64可将加密的报文转换成ASCII字符

@数据分段

- @PGP具有分段和组装功能

55

PGP安全功能列表和符号表

- @ Ks: 常规加密中的会话密钥
- @ KRa: 用户A的私有密钥
- @ KUa: 用户A的公开密钥
- @ EP: 公开密钥加密
- @ DP: 公开密钥解密
- @ EC: 常规加密
- @ DC: 常规解密
- @ H: 散列函数
- @ ||: 串接操作(并置)
- @ Z: 使用ZIP算法进行压缩
- @ R64: 基数为64的ASCII格式转换

服务	采用算法	说明
数字签名 (包含鉴别)	DSS/SHA或 RSA/SHA	用SHA-1创建散列码, 用发送者的私 钥和DSS或RSA加密消息摘要
消息加密	CAST或IDEA或 3DES、AES 及RSA或D-F	消息用一次性会话密钥加密, 会话密 钥用接收方的公钥加密
压缩	ZIP	消息用ZIP算法压缩
邮件兼容性	Radix 64	邮件应用完全透明, 加密后的消息用 Radix 64转换
数据分段		为了适应邮件的大小限制, PGP支持 分段和重组

56

PGP : 签名与认证

@安全性分析

- @RSA的强度保证了发送方的身份
- @SHA-1的强度保证了签名的有效性
- @DSS/SHA-1可选替代方案

@签名与消息可以分离

- @对消息进行单独的日志记录
- @可执行程序的签名记录, 检查病毒
- @文档多方签名, 可以避免嵌套签名

58

PGP : 机密性

@对称加密算法和公钥加密算法的结合可以缩短加密时间。

@用公钥算法解决了会话密钥的单向分发问题。

@不需要专门的会话密钥交换协议

@由于邮件系统的存储-转发的特性, 用握手方式交换密钥不太可能

@每个消息都有自己的一次性密钥, 进一步增强了保密强度。

60

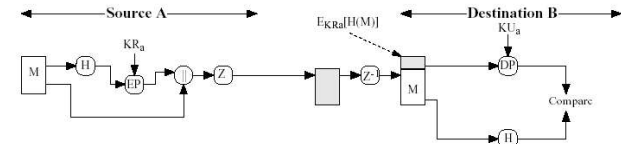
PGP : 签名与认证

@发送方

- @产生消息M
- @用SHA-1对M生成一个160位的哈希码H
- @用发送者的私钥对H加密, 并与M连接

@接收方

- @用发送者的公钥解密并恢复哈希码H
- @对消息M生成一个新的哈希码, 与H比较。如果一致, 则消息M被认证。



57

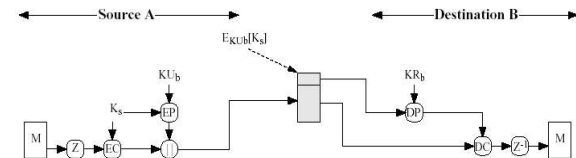
PGP : 机密性

@发送方

- @生成消息M, 并为该消息生成128位会话密钥随机数
- @采用CAST-128(或IDEA或3DES)对消息M加密
- @采用RSA, 使用接收者的公开密钥加密会话密钥, 并置到消息M前面

@接收方

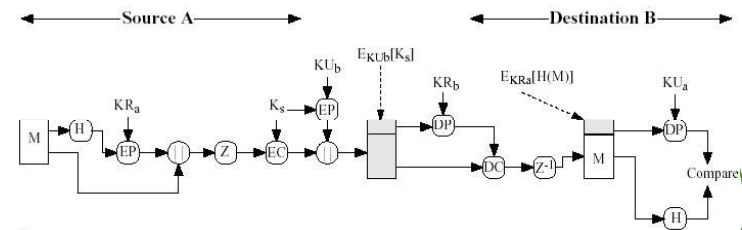
- @接收者采用RSA, 用自己的私钥解密和恢复会话密钥
- @接收者使用会话密钥解密消息M
- @也可以使用Diffie-Hellman算法生成会话密钥



59

PGP : 机密性+认证

@两种服务都需要时, 发送者先用自己的私钥签名, 然后用会话密钥加密消息, 再用接收者的公钥加密会话密钥。



61

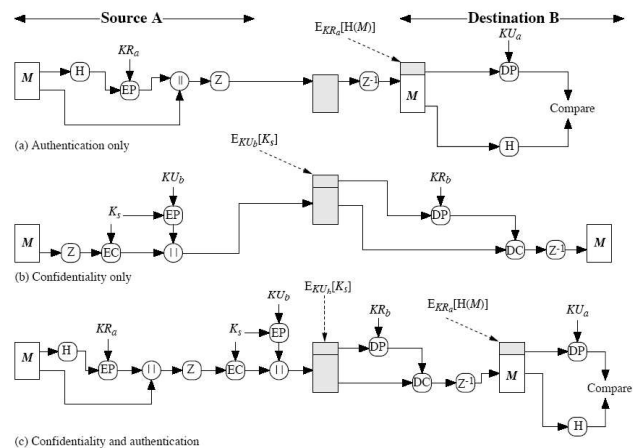


Figure 15.1 PGP Cryptographic Functions

62

PGP : 兼容性

- @PGP提供把原始8位二进制流转换成可打印ASCII字符的服务
- @采用Radix-64转换，每三个字节的二进制数据为一组映射成四个ASCII字符，附加CRC校验
- @使用Radix-64，报文长度增加了33%

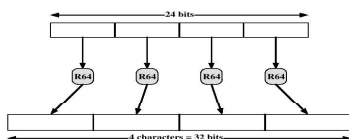


Figure 5.11 Printable Encoding of Binary Data into Radix-64 Format

64

PGP : 消息的发送与接收

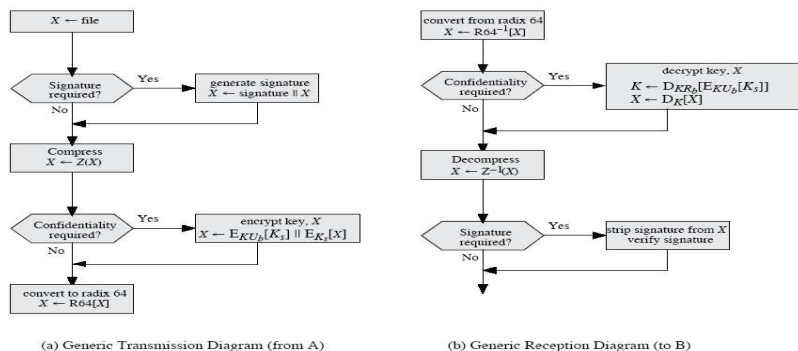


Figure 15.2 Transmission and Reception of PGP Messages

36

PGP : 压缩

- @PGP采用ZIP算法进行压缩，压缩有利于在电子邮件传输和存储时节省空间
- @压缩在签名之后进行
 - @对没有压缩过的消息进行签名，验证时只需要存储没有压缩过的消息和签名
 - @便于采用不同的压缩算法
- @加密压缩过的报文可以加强加密的强度，因为冗余减少，密码分析更加困难

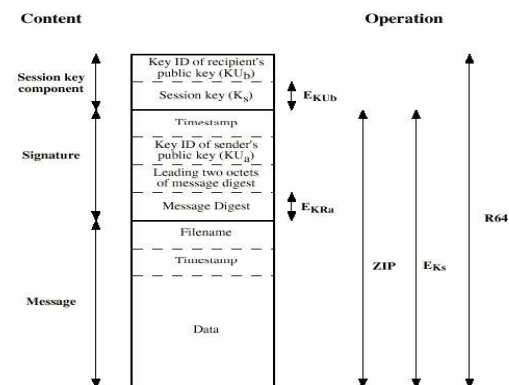
63

PGP : 分段与重组

- @电子邮件工具通常限制消息的最大长度，一般在50,000字节
- @任何大于该长度的消息必须分成若干小段，单独发送
- @PGP在发送方，自动将长消息分段使之可以通过电子邮件发送；分段在所有操作之后进行，包括Radix-64转换
- @接收方剥去所有电子邮件头，按步骤重新组装

65

PGP : 消息格式



67

小结：S/MIME与OpenPGP的对比



	S/MIME v3	OpenPGP
机密性算法	3DES系列	3DES系列
数字证书	x.509v3	早期的PGP格式
数字签名算法	DSS	DSS
认证算法	SHA-1	SHA-1

68

垃圾邮件的定义与危害

@垃圾邮件的定义：未经允许发送的出于商业宣传、政治和色情等非正常目的邮件和病毒邮件

@垃圾邮件的危害

- @大量占用网络带宽资源
- @浪费了服务器的处理资源
- @增加用户对邮件的处理时间
- @大量反动政治和色情邮件给社会带来了极大的负面影响

@垃圾邮件的泛滥是因为电子邮件服务是网络中最常用的网络服务，速度快、范围广、成本低

70

用户

@安全意识

@客户端过滤

@垃圾邮件投诉

- @查看信头：<http://www.ccert.edu.cn/spam>
- @追踪来源 whois, nslookup, traceroute, samspade.org
- @投诉：abuse@domain-name

72

反垃圾邮件常用技术

用户、邮件服务器管理员
网络服务提供商ISP、反垃圾邮件联盟

69

垃圾邮件的特点和发送手段

- @发件IP不固定
- @发件人地址不固定
- @收件人地址不固定
- @主题、内容、附件均有相对的随机性和固定性
- @时间不集中
- @通过open relay服务器转发
- @通过发送工具直接投递到邮件服务器
- @自己邮件服务器发送
- @利用ADSL、ISDN等宽带接入来发送
- @用客户端直接发送
- @通过网页邮件发送程序漏洞发送

71

服务器端反垃圾邮件：基于SMTP的检查

- @SMTP发信认证
- @MAIL FROM地址的合法性检查
- @DATA数据的合法性检查
- @RBL列表
- @收发件人地址的拒收(MAIL FROM、RCPT TO)
- @HELO/EHLO主机名检查
- @相同IP的并发连接数
- @相同IP的连接频率
- @相同帐号的发信频率(MAIL FROM)
- @SMTP认证频率
- @SMTP指令错误数

73

垃圾邮件过滤的内容

@垃圾邮件过滤的内容

@IP地址

@信封

@ Mail From, Rcpt To

@信头

@ From, To, Subject

@内容

@ 关键字: 联系信息/URL

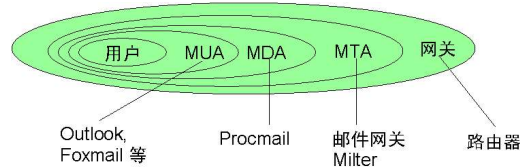
@ 邮件大小/长度

@ 群发特性及其他

@过滤方法

@ 统计, 概率

@ 规则



74

服务器端反垃圾邮件：邮件服务器配置

@关闭 Open Relay

@问题: 用户漫游

@发邮件认证

@问题: 外来的、发给本地用户的垃圾邮件

@反向域名解析

@防止动态的IP地址

@问题: 无法阻止以虚假信息注册

76

MTA过滤

邮件网关

SMTP

过滤网关

邮件服务器 MTA

MUA

过滤器

SMTP

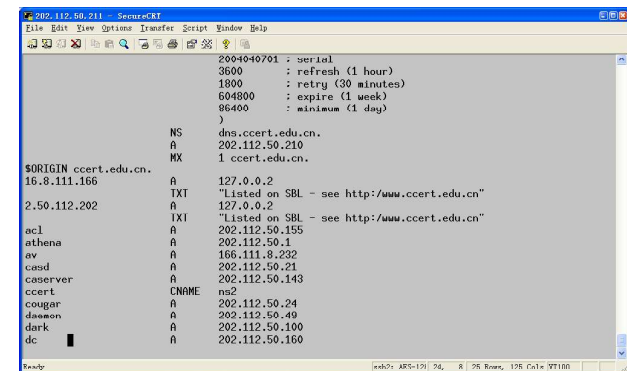
邮件服务器 MTA

过滤器

MUA

75

实时黑名单(RBL)机制

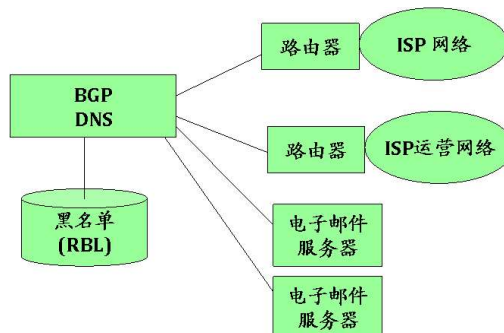


77

基于IP地址的实时过滤黑名单

@<http://mail-abuse.org>

@<http://www.spamhaus.org>



78

客户端反垃圾邮件：内容过滤



@规则过滤

@关键字概率算法判断

@收发件人拒收

@信任收发件人

@主题拒收和主题关键字拒收

@基于RBL列表的邮件头IP过滤

@邮件头信息匹配

@累计评分判断

79

课程主要内容

- @一、计算机网络安全概论
- @二、密码学基础
 - @密码学的基础知识
 - @对称密钥密码算法
 - @公开密钥密码算法
 - @密钥的分配
- @三、认证技术
 - @消息认证技术
 - @身份认证及其实例
- @四、网络安全协议
 - @网络层安全：IPsec、IKE
 - @传输层安全：SSL
 - @应用层安全：电子邮件安全、**电子商务安全**、WEB安全
- @五、各类专用技术

2

电子商务安全 需求分析和体系结构

4

网络支付面临的安全问题（1）

- @支付帐号和密码等隐私信息在网络传输中被窃取或盗用。
 - @例如：信用卡号码和密码被窃取盗用，给购物者造成损失。
- @支付金额被更改。
 - @例如：支付命令在网上发出后，从帐号中划走了与预定价格不符的经费金额。
- @支付方不能确认商家是谁，商家不能清晰确定如信用卡等网络支付工具是否真实、资金何时入帐等。
 - @例如：一些不法商家或个人利用网络贸易的非面对面性，利用互联网的开放性和不确定性，进行欺骗。

6

电子商务安全

- @需求分析和体系结构
 - @网络支付面临的安全问题
 - @网络支付的安全需求
 - @电子商务安全体系
- @安全电子交易协议SET
 - @SET协议简介
 - @SET协议参与方
 - @SET协议交易过程
 - @SET协议支付处理过程

3

电子商务安全

- @电子商务是利用信息系统和互联网完成商业贸易交易业务。
- @网络支付是电子商务的一个重要环节，快捷、方便、可靠的网络支付方式的普及应用正是体现电子商务魅力的地方。
- @网上交易必然涉及到客户、商家、银行及相关管理认证部门等多方机构及他们之间的系统配合；因为涉及到资金的划拨，更使客户和商户必须考虑网络支付结算是否安全，这也是电子商务中最重要的安全问题所在。
- @保证电子商务的安全其实很大部分就是保证电子商务过程中网络支付的快捷、方便、可靠、安全，这正是银行、商家，特别是客户关心的焦点。

5

网络支付面临的安全问题（2）

- @卖方或者买方对支付行为、内容随意抵赖、修改和否认。
 - @例如：买方当日没有支付，却坚持已经支付完毕；卖方已收到货款却矢口否认；任意一方随意改动交易金额等
- @网络支付系统故意被攻击、网络支付被故意延迟等
 - @例如：网络病毒等造成网络支付系统的错误、瘫痪、支付结算过程被故意拖延等造成客户或商家的损失等

7

网络支付的安全需求（1）

@保证网络上资金流数据的保密性

@网上交易是交易双方的事，并不希望第三方知道交易的具体情况，包括资金帐号、客户密码、支付金额等网络支付信息。

@但是，交易是在互联网上进行的，所传送的信息很容易被截获，所以必须对传送的资金数据进行加密。

@保证相关网络支付结算数据的完整性。

@数据在传送过程中不仅要求不被别人窃取，还要求数据在传送过程中不被篡改，保持数据的完整性。

@如果无法证实网上支付信息数据是否被篡改，是无法长久在网上进行交易活动的。

8

网络支付的安全需求（3）

@保证网络上资金支付结算行为发生及发生内容的不可抵赖

@在传统现金交易中，交易双方一手交钱，一手交货，没有多大问题。如果在商店里用信用卡付款，也必须持持卡人签名，方能取走货物。

@网上交易中，持卡人与商店通过网上传送电子信息来完成交易，也需要有使交易双方对每笔交易都认可的方法。

@必须为网络支付结算提供一种使交易双方在支付过程中都无法抵赖的手段，使网上交易能正常开展下去。比如数字认证、时间戳等。

@保证网络支付运行可靠、网络结算速度快捷

@实时的网络支付行为对性能要求很高

@网络支付系统对安全要求很高

10

电子商务安全体系

@网络系统安全：针对物理技术系统的安全问题

@保证网络设施的正常运行

@避免受到外界的恶意攻击

可靠安装、维护、管理
设置防火墙、防止病毒

@网络信息安全：针对商务逻辑系统的安全问题

@信息保密、信息完整

@身分认证、不可抵赖

@信息有效

加密技术
认证技术

@网络交易安全

@参与对象之间交易过程的安全，如安全套接层协议（SSL）、安全电子交易协议（SET）、公钥基础设施（PKI）

12

网络支付的安全需求（2）

@保证网络资金结算双方身份的认定，即网络支付与结算的前提必须是交易双方身份的唯一性和确定性。

@在实体店的面对面交易，营业员要检查持卡人的信用卡是否真实，核对持卡人签名等，证实持卡人的身份。持卡人亲自来到商店，看到商店真实存在。

@网上交易具有虚拟性，在网上没有方向、没有距离、也没有国界。持卡人要与网上商店进行交易，必须先确定商店是否真实。商店和银行都要担心网上购物的持卡人的信用卡是否真实有效。

@所以网上交易中，参加交易的各方，包括商户、持卡人和银行必须要采取如CA认证等措施，才能够认定对方的身份。

9

电子商务安全体系

@电子商务安全不仅仅是技术层面问题，而且是包含预防、检测、管理和制度层面在内的一整套体系的建设问题。

@电子商务对安全的基本要求

@授权合法性

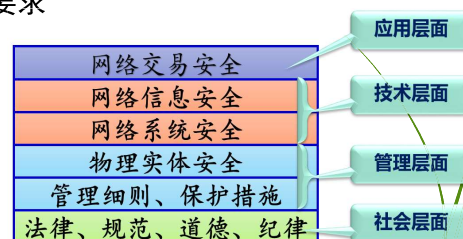
@信息的保密性

@信息的完整性

@身份的真实性

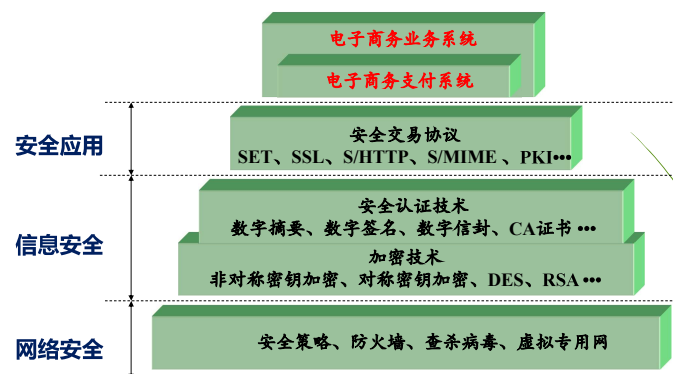
@不可抵赖性

@存储信息的安全性



11

电子商务安全技术结构示意图



13

安全电子交易协议SET

14

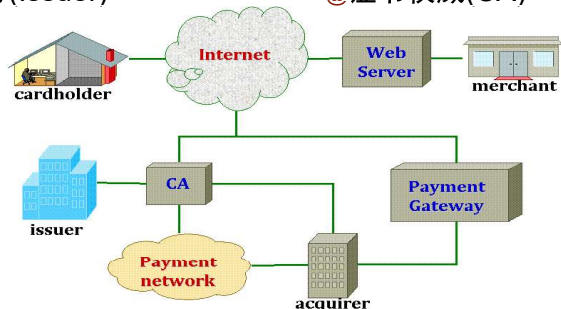
SET协议的目标

- @真实性
 - @解决多方认证问题，不仅要对消费者的信用卡认证，而且要对在线商店的信誉程度认证，同时还有消费者、在线商店与银行间的认证。
- @保密性
 - @保证信息在因特网上安全传输，防止数据被黑客或被内部人员窃取。
- @隐私
 - @保证电子商务参与者信息的相互隔离。
 - @客户的资料通过商家到达银行，但是商家不能看到客户的帐号等信息。
 - @银行不能看到用户的订单信息。
- @实时性
 - @保证网上交易的实时性，使所有的支付过程都是在线的。

16

SET协议的参与方

- @持卡人(cardholder)
- @商家(Merchant)
- @发卡方(Issuer)
- @收款行(Acquirer)
- @支付网关(Payment Gateway)
- @证书权威(CA)



18

SET协议简介

- @安全电子交易协议SET(Secure Electronic Transactions)是由世界上两大信用卡商Visa和Master Card联合制定的实现网上信用卡交易的模型和规范。
- @SET协议为在Internet上进行安全的电子商务提供了一个开放的标准，规定了交易各方进行安全交易的具体流程。
- @SET协议为持卡人、商家、银行提供了一个多方参与的安全通信信道。
- @SET协议基于X.509v3证书的身份认证，保证交易信息的私密性、保密性、完整性、抗抵赖。

15

SET协议的关键特征

- @信息的机密性
 - @卡用户的账号和支付信息在传输是保证安全。
- @数据的完整性
 - @卡用户对上网的支付信息包括定购信息，个人数据和支付指示在传输时不被修改。
- @卡用户账号的认证
 - @商人能够验证卡用户是有效卡账号的合法用户，SET用了X.509v3数字证书和RSA签名。
- @商家的认证
 - @使卡用户可以验证它可以接受支付信用卡。

17

SET协议的参与方

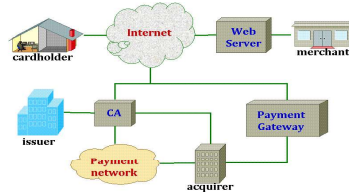
- @发卡方 (Issuer),比如银行
 - @一个金融机构，为用户和商家开户，并提供担保。
- @持卡人(cardholder)
 - @持有信用卡的消费者。
- @商家 (Merchant)
 - @一个人或一个组织，在Internet上销售产品或服务，通常向用户提供Web界面。必须事先和acquirer建立信任关系。

19

SET协议的参与方

@收款行(Acquirer)

- @商家通常接受多种信用卡，但是不愿和每个银行单独处理。
- @一个金融机构，为每个商家建立一个账号，处理每一笔交易的支付授权和实际的支付。
- @Acquirer 代替商家与多个发卡行联系，验证持卡人信用卡信息的有效性。



20

SET协议的交易过程

- @The customer opens an account
- @The customer receives a certificate
- @Merchants have their own certificates
- @The customer places an order
- @The merchant is verified
- @The order and payment information are sent
- @The merchant requests payment authorizations
- @The merchant confirms the order
- @The merchant provides the goods or service
- @The merchant requests payments

22

Step1 : 购买请求

@购买请求 (Purchase Request)

- @After cardholder browsing, selecting , ... the merchant sends a completed order to the customer
- @Cardholder → Merchant
 - @Initiate request : brand of the credit card , ID
- @Merchant → Cardholder
 - @Initiate response: signes with merchant private key, transaction ID, certificate
- @Cardholder verifies the certificate of merchant ,generates PI and OI , and places transaction ID into PI, and OI., and then , send Purchase Request Message

- @购买请求交换有四个报文组成：发起请求、发起响应、购买请求和购买响应。

24

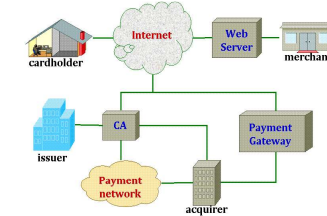
SET协议的参与方

@证书权威 (CA)

- @可信的第三方，比如国家银行
- @为cardholder, merchant , acquire 发行证书

@支付网关 (Payment Gateway)

- @由收款行操作，处理商家的支付报文。
- @支付网关是SET和银行支付网络的接口。



21

支付处理过程

@Step1: 购买请求 (Purchase Request)

- @Customer -> Merchant

@Step2: 支付授权 (Payment authorization)

- @Merchant -> payment gateway -> Issuer

@Step3: 支付获取 (Payment capture)

- @Merchant -> payment gateway -> issuer

23

Step1 : 购买请求



@Cardholder → Merchant : PI, OI

@Merchant → Acquirer : PI

- @Payment Information(PI) :Card number , money, etc.
- @Order Information(OI):Order details
- @PI 和OI 必须分开加密和签名，以保证用户的隐私不被泄漏。PI 和OI必须有联系，以防止商家篡改信息，产生纠纷。

25

双签名

@双签名的目的是为了连接两个发送给不同接收者的报文。

@PI=支付信息

@PIMD=PI报文摘要

@OI=订购信息

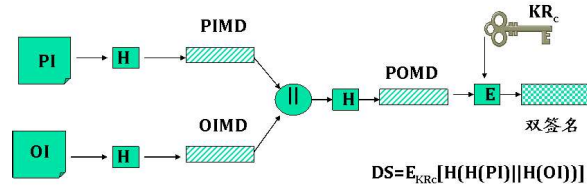
@OIMD=OI报文摘要

@H=散列函数 (SHA-1)

@E=加密 (RSA)

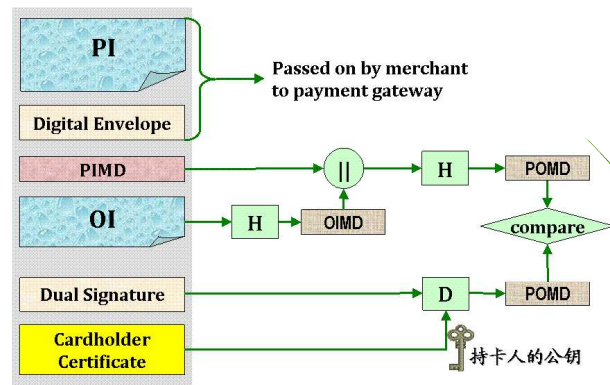
@||=拼接

@KRc=顾客的私有签名密钥



26

商家验证用户的订单



28

支付网关对支付授权请求的处理

@验证所有证书的合法性。

@解密数字信封，获得会话密钥，解密authorization block。

@验证商家对authorization block 的数字签名。

@解密Payment Block 的数字信封，解密支付信息。

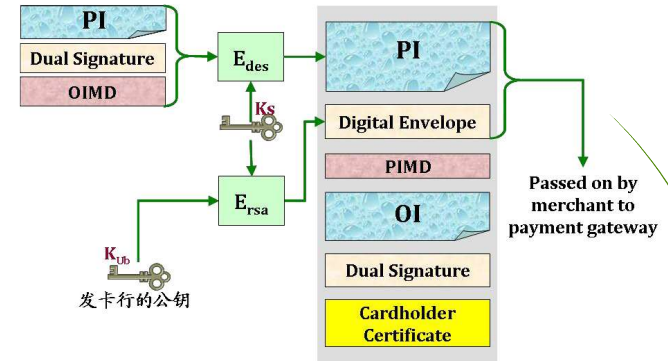
@验证双签名。

@验证transiaction ID 与PI中的是否一致。

@从发卡行申请支付。

30

客户生成购买请求



27

Step2 : 支付授权

@商家需要通过支付网关、发卡行，得到授权，才能给用户发货。

@支付授权交换由两个报文组成：授权请求和授权响应。

@授权请求报文有以下几部分组成：

@与购买有关的信息：PI, 双签名

@与授权有关的信息：Es

@证书：客户、商家

@授权响应报文有以下几部分组成：

@与授权有关的信息/获取权标信息/证书

29

Step3 : 支付获取



@商家只有通过支付获取，才能完成银行的转帐业务。

@获取请求报文由以下几部分组成：

@支付的数量

@交易ID

@获取权标

@商人的签名密钥

@证书

@支付获取由获取请求和获取响应报文组成。

@获取响应报文由以下几部分组成：

@网关的签名

@加密获取相应数据块

@网关签名密钥证书

31

课程主要内容

- @一、计算机网络安全概论
- @二、密码学基础
 - @密码学的基础知识
 - @对称密钥密码算法
 - @公开密钥密码算法
 - @密钥的分配
- @三、认证技术
 - @消息认证技术
 - @身份认证及其实例
- @四、网络安全协议
 - @网络层安全：IPsec、IKE
 - @传输层安全：SSL
 - @应用层安全：电子邮件安全、电子商务安全、**WEB安全**
- @五、各类专用技术

2

Web安全性威胁

WEB从哪里来

- @1989年3月，Tim Berners-Lee发表了一篇文章《关于信息化管理的建议》，文中提出了一个很有意思的概念：与其简单地引用其他人的著作，不如进行实际的链接；读一篇文章时，读者可以打开所引用的其他文章。
- @超文本技术(hypertext)当时相当流行，Tim Berners-Lee利用了他先前在文档和文本处理方面的研究成果，发明了超文本标记语言HTML(HyperText Markup Language)。
- @Tim Berners-Lee还创建了一个称为超文本传输协议HTTP(HyperText Transfer Protocol)的简单协议。
- @在1989年的圣诞假期，Tim Berners-Lee利用HTTP和HTML发明了第一个网页服务器和Web浏览器(同时也是编辑器)，叫做WorldWideWeb(WWW)。

6

WEB安全

- @Web安全性威胁
- @HTTP与HTTPS
 - @HTTPS=HTTP+SSL
 - @中间人攻击：ARP欺骗、DNS欺骗
- @SSL能否确保HTTPS安全?

3

WEB从哪里来

- @最早的WEB构想可以追溯到八十年代，Tim Berners-Lee是欧洲粒子物理研究中心(CERN)的研究专家，他所在研究中心的实验室分布在很多国家，大型试验往往需要很多实验室的不同试验结果协作完成。
- @但是在那个年代，没有网络，很多试验的结果要从分布在各地的实验室运到日内瓦集中处理，十分麻烦。于是Tim想创建一个平台独立和系统分布的全新信息发布系统：
 - @平台独立：使所有的用户都能访问信息发布系统而不必顾虑他们不同的机型
 - @系统分布：使得所有的用户都能够在自己的计算机上提供材料，并和其他人提供的资源联系起来

5

WEB从哪里来

- @1991年8月6日，Tim Berners-Lee在alt.hypertext新闻组上贴了WWW项目简介的文章，这一天也标志着互联网上WWW公共服务的首次亮相。
- @构成WWW的HTML文档是结构简单、平台独立的信息“容器”，这个“容器”的内容是普通文本和超文本链接。Tim Berners-Lee发明了三项WWW的基础技术：
 - @用于编写文档的超文本标记语言HTML（HTML支持超文本链接）
 - @用于发布资源的超文本传输协议HTTP
 - @通过互联网引用其他可访问文档或资源的统一资源定位URL
- @WWW改变了普通人的生活，使所有人真正进入了网络时代。

7

WEB是什么

- @WWW技术实际上是运行在互联网和TCP/IP上的一个客户/服务器程序。
- @几乎所有的商业企业、大多数的政府机构和许多个人都拥有Web站点。利用Web浏览器对互联网的访问越来越多。很多公司热衷于在Web上进行电子商务。
- @商家也渐渐意识到：互联网和Web站点实际上很容易受到攻击，因此安全的Web服务需求就应运而生了。

8

WEB的安全特点

- @很多计算机安全和网络安全技术都可以直接应用在Web中，但Web还有一些固有的安全特点。
 - @互联网是双向的。
 - @Web服务器容易受到互联网的攻击。
 - @Web越来越多地成为商业合作和产品信息发布窗口和商务交易的平台，如果Web服务器被攻击，就可能金钱失窃或信息受损。
 - @Web的使用可能比较简洁，但是底层支持软件却很复杂，容易产生安全漏洞。

10

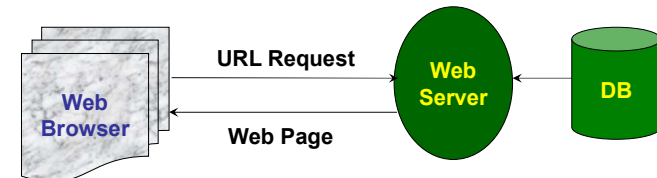
Web的安全威胁

- @对于Web的安全威胁有两类分类方法。
- @按照威胁方式分类：
 - @主动攻击包括伪装成其它用户、篡改客户和服务端之间的消息或者篡改Web站点的信息等。
 - @被动攻击包括在浏览器和服务端通信窃听、获得原来被限制使用的权限等。
- @按照威胁位置分类：
 - @Web服务器安全
 - @Web客户端安全
 - @服务器和客户端之间的通信安全

12

WEB常用技术和结构

- @HTML/XML
- @Java , Java Applet/ Servlet
- @HTTP
- @JavaScript/VBScript
- @CGI
- @ActiveX
- @Cookie



9

WEB 安全的目标

- @Web安全目标：
 - ① 确保Web服务器存储数据的安全。
 - ② 确保Web客户端的计算机是安全的。
 - ③ 确保服务器和浏览器之间的信息传输的安全。

11

Web的安全威胁

- @服务器安全
 - @用户认证A
 - @Basic
 - @MD
 - @SSL
 - @访问控制A
 - @Access
 - @GET, POST, Update
 - @日志
 - @Access_log
 - @Error_log
- @客户浏览器的安全
 - @对服务器的认证
 - @访问控制机制和签名
 - @Java Applet的本地资源访问控制
 - @ActiveX 的数字签名
- @服务器和客户端之间的通信安全
 - @IPsec
 - @SSL/TLS
 - @MIME/PGP/SET等

13

Web的安全威胁

	威胁	后果	对策
完整性	* 修改用户权限 * 特洛伊木马浏览器 * 内存修改 * 传送中的消息修改	* 信息丢失 * 机器受损 * 易受所有其它威胁的攻击	加密的校验和
保密性	* 网络窃听 * 盗窃服务器数据 * 盗窃客户端数据 * 盗窃洛网络配置信息 * 盗窃客户端和服务端通话信息	* 信息失窃 * 秘密失窃	加密、Web 代理
拒绝服务	* 破坏用户线程 * 用假消息使得机器溢出 * 装满硬盘或者内存 * 使用 DNS 攻击来孤立机器	* 破坏 * 干扰 * 阻止正常工作	难于防止
认证	* 伪装成合法用户 * 伪造数据	* 用户错误 * 相信虚假信息	加密技术

14

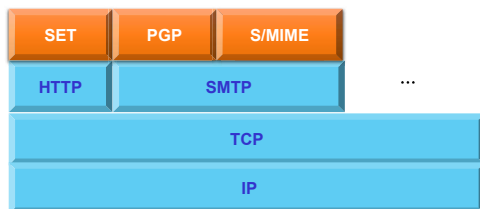
Web流量安全方法



@特定的安全服务在特定的应用实现中，即**应用级安全**。

@这种方法的好处是为应用定制的。

@Web安全的一个典型应用就是安全电子交易SET (Security Electronic Trade)。



16

超文本传送协议HTTP

@超文本传送协议HTTP(Hyper Text Transfer Protocol): HTTP是目前使用最为广泛的应用层协议，它详细规定了浏览器和WWW服务器之间互相通信的规则，使得用户能够通过浏览器看到网页里丰富的图文内容。

@HTTP的缺陷

@明文传输，没有数据完整性校验

@无状态链接，无法验证双方身份的真实性

18

Web流量安全方法

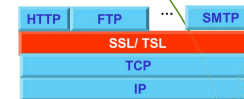
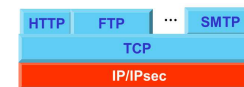
@提供Web安全的一种方法是**IP级安全**。

@使用IPSec的优点在于它对终端用户和应用都是透明的，提供通用的解决方案，而且还具有过滤功能。

@提供Web安全的另外一种方法是**TCP级安全**。

@最先使用的是SSL (Security Socket Layer)，接着是传输层安全协议TLS (Transport Layer Security)。

@SSL或TLS作为下层协议可以对应用透明，也可以在特定包中使用。



15

HTTP与HTTPS

17

常见的针对HTTP的攻击

@监听嗅探

@由于HTTP采用明文信息传输，可以被直接嗅探到明文。

@篡改劫持

@攻击者可以修改通信数据包，达到篡改信息和劫持回话的目的。

@伪造服务器

@由于HTTP协议并不验证服务器的可信度，故而存在ARP欺骗、DNS欺骗以及钓鱼等风险。

19

致命的中间人攻击

@中间人攻击MITM（Man-In-The-Middle attack）指攻击者与通讯双方分别建立独立的联系，并交换其所收到的数据。

② 通讯双方都认为他们正在通过一个私密的连接与对方直接对话，但事实上整个会话都被攻击者完全控制，而且攻击者还可以拦截通讯双方的通话并插入新的内容。

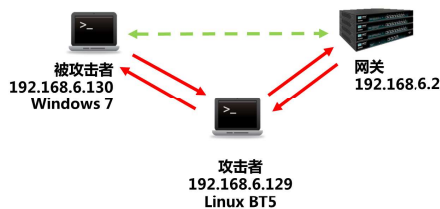
@因为HTTP协议本身采用明文传输通信内容，并且对通信双方不做验证，完全无法防御中间人攻击。

@HTTP协议无法保证通信的保密性、一致性和完整性。通信内容可以被窃取、篡改，通信目标可以被仿冒。

@ARP欺骗(ARP Spoofing)就是一种中间人攻击的常用手段。

ARP 欺骗

@ARP欺骗是一种中间人攻击的常用手段。攻击者可以通过制造伪造的ARP frame (request, reply)，修改网内任何计算机的映射表，从而切断目标主机(victim)的互联网通讯,窃取目标主机(victim)关键信息。



ARP欺骗

@使用arp spoof分别对目标主机(192.168.6.130)和网关(192.168.6.2)进行欺骗

[illegible]

地址解析协议ARP

@地址解析协议ARP (Address Resolution Protocol) :
工作在数据链路层, 实现MAC地址与IP地址的映射

@ARP协议通过建立和维护IP-MAC映射表进行工作:

@ A知道B的IP地址，在映射表中查不到B的MAC地址，就广播需求：“谁知道IP地址为B的MAC地址，请告知”

@ B发现这个请求，先将A的“IP-MAC对”存入自己的映射表，再回复A：“IP地址为B的MAC地址是*****”

@ A 接到B的回复后，将B的“IP-MAC对”存入自己的映射表

@问题：在A、B之间没有任何验证，无条件信任发来的信息，更新通讯映射表

ARP欺骗

@缓存表中网关（192.168.6.2）的MAC地址已经变成了攻击者（192.168.6.129）的MAC地址，表明ARP毒化成功

```

C:\Windows\system32\cmd.exe
C:\Users\xzhzbjng-wjn7>arp -a

接口: 192.168.6.130 --- 0x0
Internet 地址          物理地址              类型
192.168.6.1            00-50-56-c0-00-08     动态
192.168.6.2            00-50-56-e5-df-06     静态
192.168.6.129          00-0c-27-0e-12-83     静态
192.168.6.254          00-0c-29-0e-12-83     静态
192.168.6.255          00-50-56-a2-11-11     静态
192.168.6.255          ff-ff-ff-ff-ff-ff     静态
224.0.0.22             01-00-5a-00-00-16     静态
224.0.0.252            01-00-5a-00-00-fc     静态
239.255.255.250        01-00-5e-7f-ff-fa     静态
255.255.255.255        ff-ff-ff-ff-ff-ff     静态

C:\Users\xzhzbjng-wjn7>arp -a

接口: 192.168.6.130 --- 0x0
Internet 地址          物理地址              类型
192.168.6.1            00-50-56-c0-00-08     动态
192.168.6.2            00-0c-27-0e-12-83     静态
192.168.6.129          00-0c-27-0e-12-83     静态
192.168.6.254          00-50-56-a2-11-11     静态
192.168.6.255          ff-ff-ff-ff-ff-ff     静态
224.0.0.22             01-00-5a-00-00-16     静态
224.0.0.252            01-00-5a-00-00-fc     静态
239.255.255.250        01-00-5e-7f-ff-fa     静态
255.255.255.255        ff-ff-ff-ff-ff-ff     静态

```

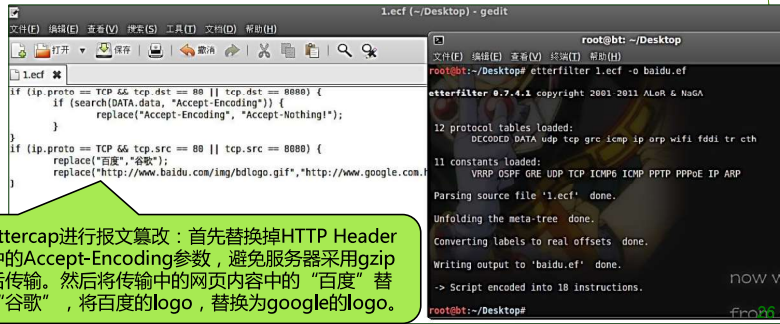
ARP欺骗

@在攻击者的电脑上打开wireshark对本机网卡进行抓包，可以抓取到被攻击者的HTTP请求，并且能获取到明文的密码等敏感信息。

The screenshot displays the Wireshark 1.8.3 interface with a packet capture of an HTTP GET request. The packet list shows a GET request to /static/info/common/check_right.php. The packet details pane shows the request structure, including the Host header '192.168.0.130' and the User-Agent 'Mozilla/5.0 (Windows NT 6.0; rv:3.0) Gecko/20100101 Firefox/3.0'. The packet bytes pane shows the raw data of the request.

ARP欺骗

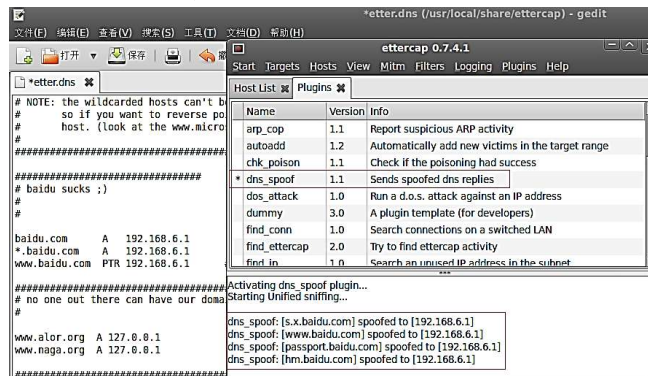
@当被攻击者的流量从本机网卡转发时，攻击者不仅可以获取到敏感信息，更可以对通信内容进行篡改。



使用ettercap进行报文篡改：首先替换掉HTTP Header请求中的Accept-Encoding参数，避免服务器采用gzip压缩后传输。然后将传输中的网页内容中的“百度”替换为“谷歌”，将百度的logo，替换为google的logo。

DNS欺骗

@ARP欺骗成功后，可以继续继续进行DNS欺骗。



28

安全的HTTP : HTTPS

@HTTPS是HTTP和SSL/TSL协议的合并，解决了数据加密、完整性校验、对服务器的身份认证等问题。

@HTTPS和HTTP在安全方面区别包括：

@HTTP协议的信息是明文传输的，HTTPS协议则采用传输层的SSL协议加密传输信息。

@HTTP和HTTPS采用完全不同的连接方式，端口也不一样，HTTP端口号为80，HTTPS端口号为443。

@HTTP协议是简单的无状态连接；HTTPS协议是由SSL+HTTP协议构建的可进行加密传输、身份认证的连接。

30

ARP欺骗

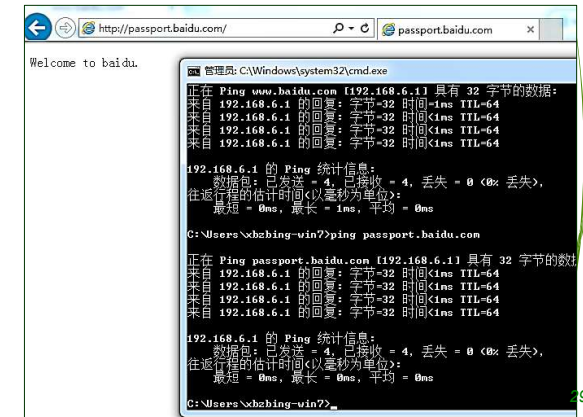
@被ARP毒化后的主机，再访问百度主页，就出现了通信数据已经被篡改的现象。



27

DNS欺骗

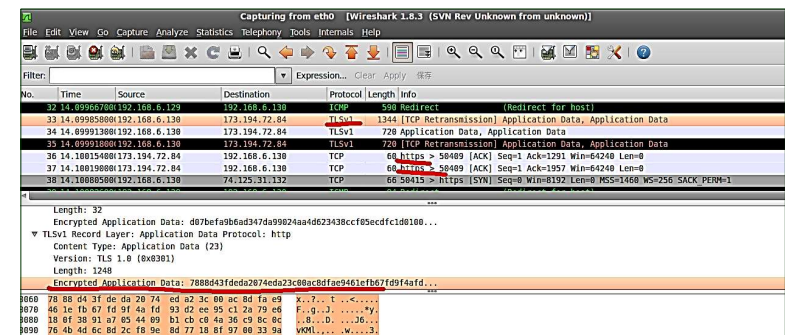
@DNS欺骗成功



29

HTTPS与ARP欺骗

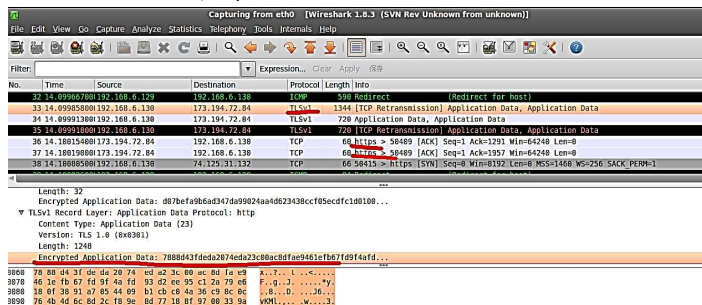
@使用HTTPS协议依旧无法避免ARP欺骗和嗅探攻击，但是由于HTTPS采用SSL加密传输，即时被嗅探到也无法得到明文信息。



31

HTTPS与报文篡改

@使用ettercap进行报文的篡改，首先要禁止服务器对通信内容进行压缩，因为压缩后内容变为乱码，无法识别。使用HTTPS后，内容不压缩也是无法识别的密文，攻击者无法对通信内容进行篡改。

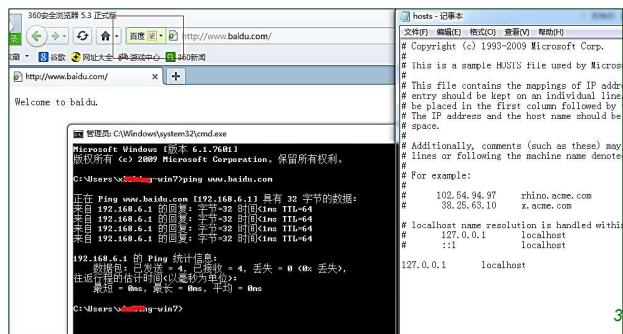


32

HTTP与HTTPS的对比



@HTTP无法验证服务器身份，一些浏览器看起来会做一些网站身份验证，不过这种做法是基于网站域名的，当遇到DNS劫持/欺骗时完全无法抵抗。



34

HTTPS = HTTP+SSL

- @用户：浏览器里输入 https://www.sslserver.com
- @HTTP层：将用户需求翻译成HTTP请求
- @SSL层：借助下层协议的信道安全的协商出一份加密密钥，并用此密钥来加密HTTP请求。
- @TCP层：与web server的443端口建立连接，传递SSL处理后的数据。接收端与此过程相反。
- @SSL在TCP之上建立了一个加密通道，通过这一层的数据经过了加密，因此达到保密的效果。

36

HTTPS与服务器认证

@采用HTTPS通信时，浏览器地址栏会有一个小锁，点击会显示网站标识。由于HTTPS的加密证书是在CA（证书管理机构）申请后发放，所以拥有该证书的单位和个人是唯一的、不可仿冒的。



33

SSL能否确保HTTPS安全?



HTTPS与证书替换

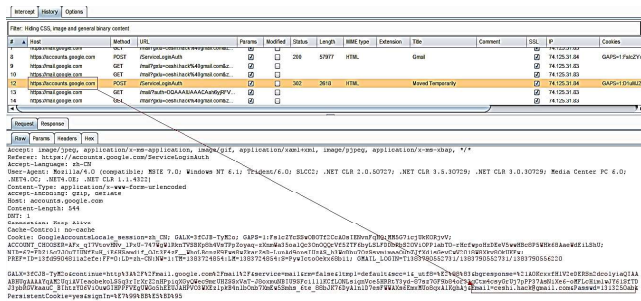
@当用户通过钓鱼WiFi或者恶意代理访问HTTPS网站时，服务器的证书可能被替换掉了。替换后浏览器会用红色表示地址栏并提示证书错误、不受信任的证书，但是如果用户依然在这种环境下使用网络，则其信息可能就会被盗取。



37

HTTPS与证书替换

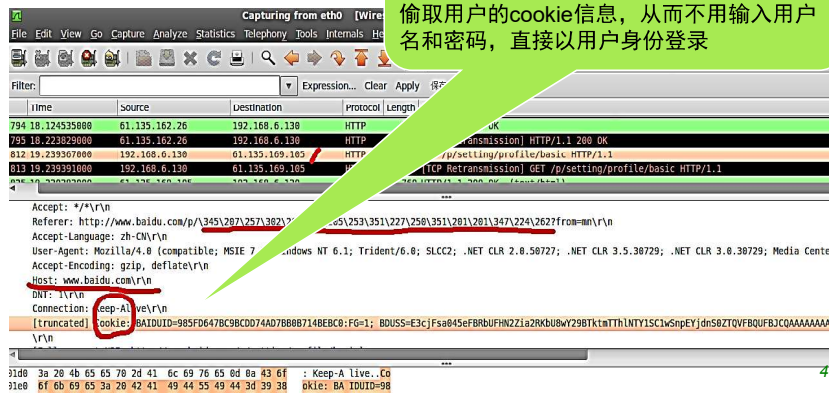
@攻击者使用自己的证书替换掉服务器的证书，那么通信过程就是用攻击者的证书进行加密，则对攻击者来说，就是明文通信。



38

会话劫持

攻击者无法截取到登录的用户名和密码，但是在登录之后会跳转到HTTP页面，此时的会话保持采用的是cookie验证。攻击者可以偷取用户的cookie信息，从而不用输入用户名和密码，直接以用户身份登录



40

SSLStrip : 针对HTTPS的攻击

@2009年BlackHat大会上，名为Moxie Marlinspike的黑客上公布了SSLStrip工具

@SSLStrip的工作原理及步骤如下：

@先进行中间人攻击来拦截 HTTP 流量

@将出现的 HTTPS 链接全部替换为 HTTP，同时记下所有改变的链接

@使用 HTTP 与受害者机器连接

@同时与合法的服务器建立 HTTPS

@受害者与合法服务器之间的全部通信经过了代理转发

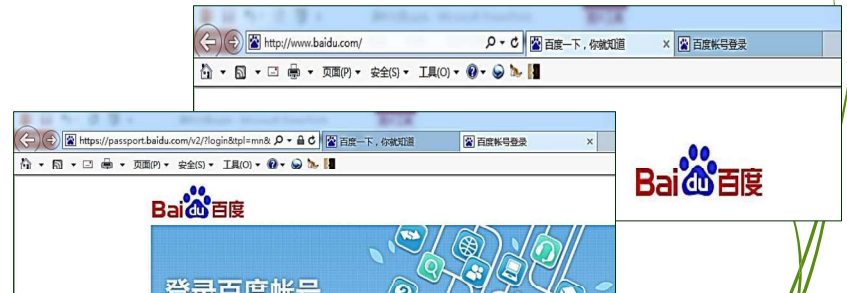
@其中出现的图标被替换成为用户熟悉的“小黄锁”图标，以建立信任

@这样中间人攻击就成功骗取了密码、账号等信息，而受害者一无所知

42

HTTPS与会话劫持

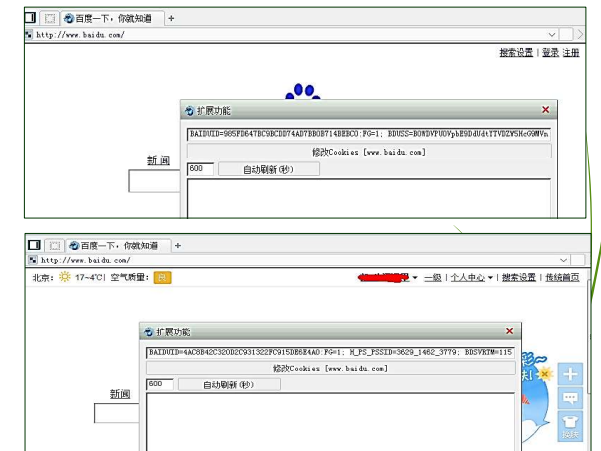
@以百度为例，整个通信过程并不全是基于HTTPS协议，只有在登录时采用HTTPS传输用户的用户名和密码。



39

会话劫持

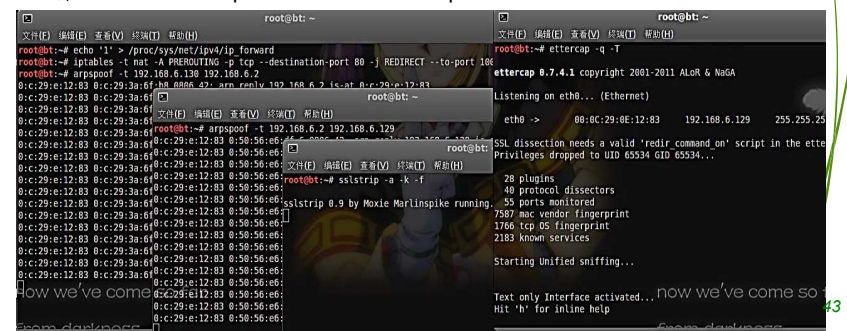
@使用被攻击者的cookie登录百度。



41

SSLStrip : 针对HTTPS的攻击

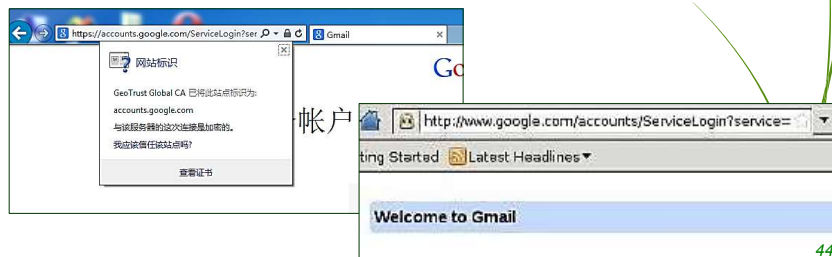
@开启系统的转发功能，并使用iptables将80端口的数据转发给本机的10000端口（sslstrip默认使用10000端口），然后使用arpspoof对目标主机和网关进行arp欺骗，同时开启sslstrip。这时候使用ettercap嗅探本机网卡数据。



43

SSLStrip : 针对HTTPS的攻击

@仔细观察二者的差异：使用sslstrip后，其实通信已经变成http了，通信是明文的，可以直接嗅探。地址栏出现了一个小锁图标，是用来迷惑被攻击者，以为还是在https协议下。这样还避免了替换证书时，浏览器的红色警告，被攻击者如果安全意识不强，就中招了。



SSLStrip : 针对HTTPS的攻击



@嗅探得到
Google帐号的
登录名和密码

```
root@bt:~# ettercap -q -T
ettercap 0.7.4.1 copyright 2001-2011 ALoR & NaGA
Listening on eth0... (Ethernet)

eth0 -> 00:0C:29:0E:12:83 192.168.6.129 255.255.255.0

SSL dissection needs a valid 'redir command on' script in the etter.conf file
Privileges dropped to UID 65534 GID 65534...

 28 plugins
 40 protocol dissectors
 55 ports monitored
7587 mac vendor fingerprint
1766 tcp OS fingerprint
2183 known services

Starting Unified sniffing...

Text Only Interface activated...
Hit 'h' for inline help

SEND L3 ERROR: 1526 byte packet (0800:06) destined to 123.125.82.133 was not forwarded (libnet_write_raw_ipv4(): -1 bytes written (Message too long))

HTTP/3.1273.194.72.84:80 -> USER: ceshi.hack@gmail.com PASS: 1313250ab INFO: http://accounts.google.com/ServiceLogin?service=mail&passive=true&rm=false&continue=http://mail.google.com/mail/6scc=1&tmpl=default&tmpicache=2&emr
```

课程主要内容

- @一、计算机网络安全概论
- @二、密码学基础
 - @密码学的基础知识
 - @对称密钥密码算法
 - @公开密钥密码算法
 - @密钥的分配
- @三、认证技术
 - @消息认证技术
 - @身份认证及其实例
- @四、网络安全协议
 - @网络层安全：IPsec、IKE
 - @传输层安全：SSL
 - @应用层安全：电子邮件安全、电子商务安全、WEB安全
- @五、各类专用技术
 - @访问控制技术

2

访问控制模型

4

访问控制的基本概念

- @主体(Subject)
 - @主体是一个主动的实体，它提出对资源访问请求。如用户，程序，进程等
- @客体(Object)
 - @含有被访问资源的被动实体，如网络、计算机、数据库、文件、目录、计算机程序、外设、网络
- @访问(Access)
 - @对资源的使用，读、写、修改、删除等操作，例如访问存储器；访问文件、目录、外设；访问数据库；访问一个网站

6

访问控制技术

- @访问控制模型
 - @访问控制的基本概念
 - @基本的访问控制政策模型
- @防火墙的基本概念
- @防火墙技术
 - @包过滤技术、地址转换、电路层网关、应用层代理
- @防火墙的配置结构
- @虚拟局域网技术VLAN

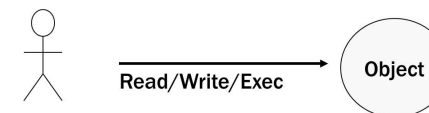
3

访问控制的基本概念

5

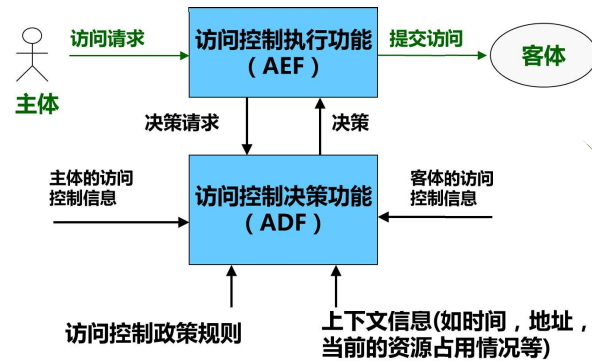
访问控制的基本概念

- @访问可以被描述为一个三元组 (s, a, o)
 - @主体、发起者：Subject、Initiator
 - @客体、目标：Object、Target
 - @访问操作：Access



7

访问控制模型



8

访问控制的基本概念

@访问控制信息(ACI)的表示

- @主体访问控制属性
- @客体访问控制属性
- @访问控制政策规则

@授权(Authorization)

- @怎样把访问控制属性信息分配给主体或客体
- @如何浏览、修改、回收访问控制权限

9

访问控制矩阵

@按列看是客体的访问控制列表

- @access control list

@按行看是主体的访问能力表

- @capability list

Subjects	Objects		
	O ₁	O ₂	O ₃
S ₁	Read/write		
S ₂		Write	
S ₃	Execute		Read

10

访问控制矩阵

@访问控制机制可以用一个三元组来表示(S,O,M)

- @主体的集合 $S=\{s_1, s_2, \dots, s_m\}$
- @客体的集合 $O=\{o_1, o_2, \dots, o_n\}$
- @所有操作的集合 $A=\{R, W, E, \dots\}$
- @访问控制矩阵

$$M = S \times O \rightarrow 2A$$

$$M = \begin{bmatrix} a_{00} & a_{01} & \dots & a_{0n} \\ a_{10} & a_{11} & \dots & a_{1n} \\ \dots & \dots & \dots & \dots \\ a_{m0} & a_{m1} & \dots & a_{mn} \end{bmatrix}$$

对于任意 $s_i \in S, o_j \in O$, 存在 $a_{ij} \in M$ 决定 s_i 对 o_j 允许的操作。比如 $a_{ij} = \{R, W\}, a_{ik} = \phi$

11

访问控制矩阵

@矩阵第i行Si表示了主体si对所有客体的操作权限，称为主体si的能力表(Capability List)

@矩阵第j列Oj表示客体oj允许所有主体的操作，称为oj的访问控制表(ACL)

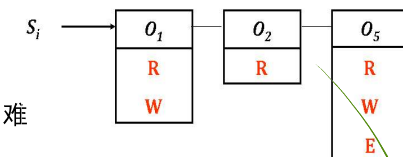
$$M = \begin{bmatrix} a_{00} & a_{01} & \dots & a_{0n} \\ a_{10} & a_{11} & \dots & a_{1n} \\ \dots & \dots & \dots & \dots \\ a_{m0} & a_{m1} & \dots & a_{mn} \end{bmatrix} = \begin{bmatrix} S_0 \\ S_1 \\ \dots \\ S_m \end{bmatrix} \times [O_0 \ O_1 \ \dots \ O_n]$$

12

能力表(Capability List)

@能力表与主体关联，规定主体所能访问的客体和权限。

- @从能力表得到一个主体所有的访问权限很容易
- @从能力表浏览一个客体所允许的访问控制权限很困难



@表示形式：用户Profile

- @由于客体相当多，分类复杂，不便于授权管理（通过授权证书、属性证书等）

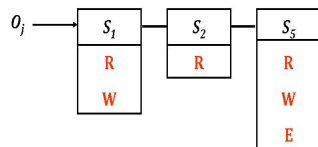
13

访问控制表(Access Control List)

@访问控制表与客体关联，规定能够访问它的主体和权限

@得到一个客体所有的访问权限很容易

@浏览一个主体的所有访问权限很困难



@由于主体数量一般比客体少得多而且容易分组，授权管理相对简单

14

访问控制与其他安全机制的关系

@身份认证

@是访问控制的前提，保证主体身份的真实性

@保密性

@限制用户对数据的访问（读取操作），可以实现数据保密服务

@完整性

@限制用户对数据的修改，实现数据完整性保护

@可用性

@限制用户对资源的使用量，保证系统的可用性

@安全管理相关的活动

@访问控制功能通常和审计、入侵检测联系在一起

16

基本的访问控制政策模型

@自主型访问控制(DAC)

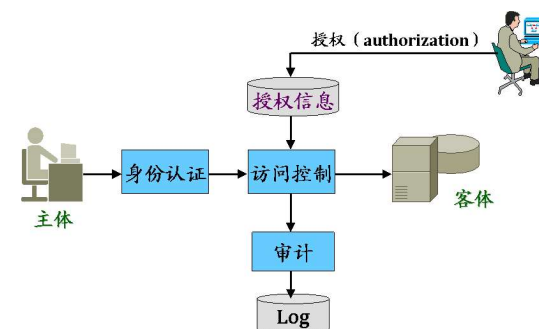
@强制型访问控制(MAC)

@基于角色的访问控制(RBAC)

18

访问控制与其他安全机制的关系

@认证、授权、审计 (AAA)



15

基本的访问控制政策模型

17

DAC：自主型访问控制

@DAC: Discretionary Access Control

@DAC是一种分布式授权管理的模式

@每个客体有一个属主，属主可以按照自己的意愿把客体的访问控制权限授予其他主体

@DAC控制灵活，易于管理，是目前应用最为普遍的访问控制政策

@DAC无法控制信息流动

@信息在移动过程中，其访问权限关系会被改变

@如用户A可将其对目标O的访问权限传递给用户B，从而使不具备对O访问权限的B可访问O

19

举例：Unix 文件系统

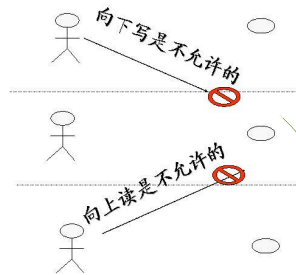
```
/bin/ls
[root@acl tmp]# chown root ls
[root@acl tmp]# ls -l
-rw-r--r-- 1 nobody nobody 770 Oct 18 15:16 4011.tmp
-rw----- 1 root users 48 Oct 28 11:41 ls
srwxrwxrwx 1 root root 0 Aug 29 09:04 mysql.sock
drwxrwxr-x 2 duan uan 4096 Oct 23 23:41 ssl
[root@acl tmp]# chmod o+rw ls
[root@acl tmp]# ls -l
-rw-r--r-- 1 nobody nobody 770 Oct 18 15:16 4011.tmp
-rw-----rw- 1 root users 48 Oct 28 11:41 ls
srwxrwxrwx 1 root root 0 Aug 29 09:04 mysql.sock
drwxrwxr-x 2 duan duan 4096 Oct 23 23:41 ssl
[root@acl tmp]#
```

20

MAC：强制型访问控制

@No Read Up:

- @Subject只能读取安全级别小于或等于自己的客体



@No write down:

- @主体只能写入到安全级高于或等于自己高的Object
- @Property

22

RBAC：基于角色的访问控制

@用户组(group): 用户的集合G

- @G={s1, s2, s3 ...}的授权管理包括
 - @定义用户组
 - @把某个资源的访问权限分配给某个用户组
 - @把用户分配到某个用户组

@角色(Role): 完成任务须访问的资源及相应操作权限的集合R

- @R={(a1,o1), (a2,o2), (a3,o3)...}的授权管理包括
 - @根据任务需要定义角色
 - @为角色分配资源和操作权限
 - @给一个用户指定一个角色

24

MAC：强制型访问控制

@MAC: Mandatory Access Control

- @每个主体和客体分配一个固定的安全级别，只有系统管理员才可以修改

- @Clearance, classification, sensitivity
- @Unclassified<confidential<secret<top secret
 - @普密 <秘密 <机密 <绝密

- @只有在主体和客体的安全级别满足一定规则时才允许访问

21

MAC：强制型访问控制的信息流

Subjects					Objects
R/W	W	W	W	☐	TS
R	R/W	W	W	☐	S
R	R	R/W	W	☐	C
R	R	R	R/W	☐	U
TS	S	C	U		

← 信息流向

23

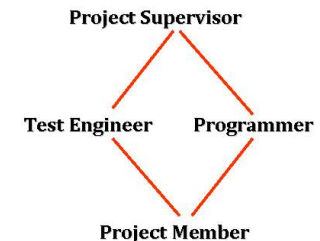
RBAC：基于角色的访问控制



@角色的继承关系

@RBAC的优势

- @便于授权管理
- @便于责任划分



@参考文献:

- @Ravi S. Sandhu Role-Base Access Control Model, Computer, Feb. 1996

25

防火墙的基本概念

26

防火墙的由来

@说法一：“防火墙”这个术语参考来自应用在建筑结构里的安全技术

@在楼宇里用来起分隔作用的墙，用来隔离不同的公司或房间，尽可能的起防火作用

@说法二：“防火墙”是汽车一个部件的名称

@在汽车中，利用防火墙把乘客和引擎隔开，汽车引擎一旦着火，防火墙不但能保护乘客安全，而同时还能让司机继续控制引擎

28

防火墙的定义

@防火墙是在被保护网络和其他网络之间限制访问的一种设备

@由软件和硬件设备组成、在内部网和外部网之间、专用网与公共网之间的界面上构造的保护屏障

@防火墙的作用

- @隐藏内部网络结构及资源
- @保护不安全的网络服务
- @执行网络间的访问控制策略
- @统一集中的安全管理
- @记录并统计网络使用情况
- @监视和预警

30

防火墙的基本概念

@防火墙是在两个网络(通常是用户内部网络和Internet)之间实施访问控制政策的一个或一组系统（硬件、软件的组合）。

@为什么需要防火墙？

- @企业的业务对互联网需求
- @企业网和Internet的连接的风险
- @企业网内部网不同安全需求的网段之间的隔离



27

防火墙的设计目标

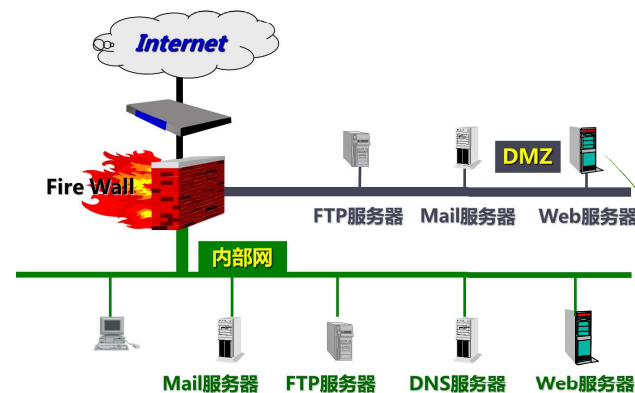
@所有的通信，无论是从内部到外部还是从外部到内部的，都必须经过防火墙。

@只有授权的通信才能通过防火墙，这些授权将在本地安全策略中规定。不同类型的防火墙将实现不同的安全策略。

@防火墙本身对于渗透必须是免疫。这意味着必须使用运行安全操作系统可信系统。

29

防火墙的定义



31

防火墙可以做什么

- @如果所有出入内部网络的流量都必须经过防火墙，而且防火墙自身是免疫的，那么
 - @防火墙可以在网络边界实施访问控制政策
 - @防火墙可以监视、记录所有的访问
 - @防火墙可以隐藏内部网络，例如地址转换
 - @防火墙可以进行信息认证和加密
- @防火墙定义了单一阻塞点
 - @它使得未授权的用户无法进入网络，限制了潜在的易受攻击的服务进入或是离开网络，同时防止了各种形式的IP欺骗和路由攻击

32

防火墙不可以做什么



- @防火墙需要用户定义访问控制规则，没有缺省配置
- @防火墙不能防止内部恶意的攻击者
- @防火墙无法控制没有经过它的连接
- @防火墙不能很好地防止病毒和信息扩散
- @防火墙不能替代内部网络系统的安全管理
- @防火墙无法防范全新的威胁和攻击

34

防火墙的常用技术

- @防火墙常用来控制访问和加强站点安全策略的四项技术：
 - @服务控制(Service Control)
 - @基于服务端口号，基于服务器的IP地址等
 - @方向控制(Direction Control)
 - @比如，内部发起的可以通过，但反之不允许
 - @用户控制(Service Control)
 - @基于用户的身份进行控制，特别是允许外部用户访问内部网络，如VPN
 - @行为控制(Service Control)
 - @基于用户的行为进行控制，如Spam Email filter

36

防火墙可以做什么

- @防火墙提供了一个监控安全事件的地点
 - @安全检查和警报可以在防火墙系统上实施
- @防火墙还是一个便利的平台
 - @提供了一些于网络安全无关的功能，例如地址转换器、网络管理等
 - @还可以作为IPSec的平台，利用隧道模式实现虚拟专用网络

33

防火墙技术

35

防火墙实现原理

- @防火墙对流经它的网络流量进行扫描，这样能够过滤掉一些攻击，以免其在目标计算机上被执行
- @防火墙可以关闭不使用的端口，禁止特定端口的特定流量，封锁特洛伊木马
- @防火墙可以禁止来自特殊站点的访问，从而防止来自不明入侵者的所有流量

37

防火墙的分类

- @防火墙有多种分类方式
 - @结构、性能、应用部署位置、技术
- @从防火墙结构上划分
 - @单一主机防火墙
 - @路由器集成式防火墙
 - @分布式防火墙
- @从防火墙性能上划分
 - @百兆级防火墙
 - @千兆级防火墙
 - @万兆级防火墙

38

防火墙的分类

- @从防火墙应用部署的位置上划分
 - @个人防火墙
 - @安装于单台主机中，防护的也只是单台主机
 - @主要应用于个人用户，通常为软件防火墙，价格最便宜，性能也最差
 - @边界防火墙
 - @最为传统的防火墙，对内、外部网络实施隔离，保护边界内部网络
 - @一般都是硬件类型的，价格较贵，性能较好
 - @芯片级防火墙
 - @使用专用的ASIC(专用集成电路)芯片处理能力强，性能高，价格也最贵
 - @比较知名的厂商有NetScreen、FortiNet、Cisco等

39

防火墙分类

- @从防火墙技术上划分为四类
 - @包过滤技术(Packet filtering/screening)
 - @地址转换(NAT)
 - @电路层网关(Circuit Gateway)
 - @应用层代理(Proxy)



40

包过滤技术

41

包过滤技术

- @包过滤技术工作在网络层；包过滤路由器依据一套规则对收到的IP包进行处理，决定是转发还是丢弃
- @包过滤技术是一种通用、廉价和有效的安全手段，经历了静态和动态两代
 - @第一代：静态包过滤类型防火墙
 - @第二代：动态包过滤类型防火墙
 - @Checkpoint防火墙和美国Cisco公司的PIX防火墙为包过滤技术防火墙的代表产品

42

第一代静态包过滤类型防火墙

- @这类防火墙几乎是与路由器同时产生的，它是根据定义好的过滤规则审查每个数据包，以便确定其是否与某一条包过滤规则匹配
- @过滤规则基于数据包的报头信息进行，基于以下信息对经过的每一个包进行检查：
 - @IP 源地址和目标地址
 - @协议 (TCP, UDP, ICMP, BGP等)
 - @TCP/UDP 源端口号和目标端口号
 - @ICMP的消息类型

43

第二代动态包过滤类型防火墙

@基于包状态监测技术，动态设置包过滤规则；对通过其建立的每一个连接都进行跟踪，根据需要可动态地在过滤规则中增加或更新条目



44

路由器中的访问控制列表

@路由器总是自顶向下顺序检查所有规则，配置规则时要从具体到一般

@deny 192.168.2.0/24

@permit 192.168.2.1

@permit any

@常发生的放在列表的前面

@隐含拒绝一切deny all

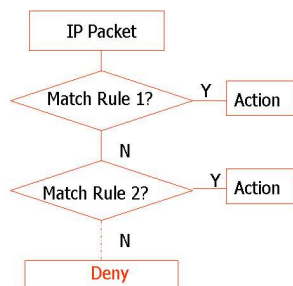
@新增加的行将放在末尾

@未定义访问控制列表等于permit all

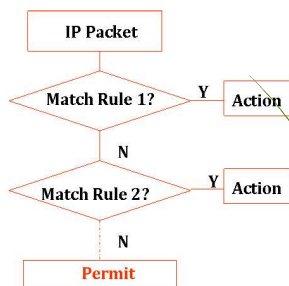
46

举例：缺省策略

@Deny Any



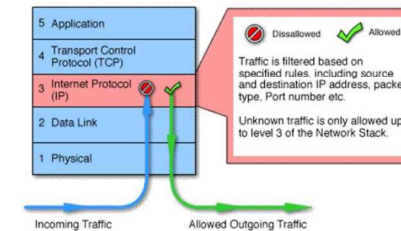
@Permit Any



48

包过滤技术

@常见包过滤设备/软件：路由器访问控制表ACL

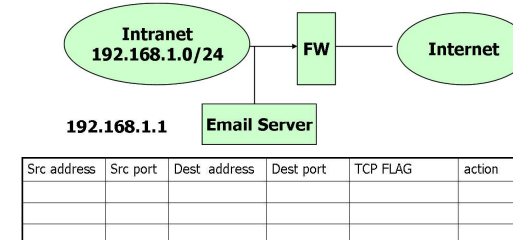


45

举例：路由器中的访问控制

@访问控制需求

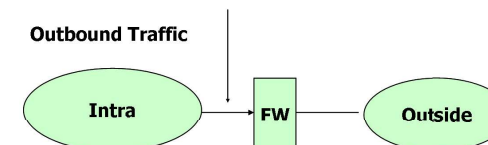
@内部用户只允访问Internet的Web服务和电子邮件；外部用户只能向内部用户发送电子邮件



47

举例：流出数据包过滤

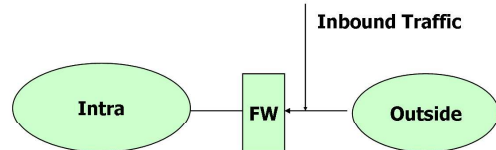
Src address	Src port	Dest address	Dest port	FLAG	action
192.168.1.0/24	any	any	80		permit
192.168.1.0/24	any	any	443		permit
192.168.1.0/24	any	any	25		permit
any	any	any	any		deny



49

举例：流入数据包过滤

Src address	Src port	Dest address	Dest port	FLAG	action
any	any	Email-Server	25	*	permit
any	any	any	any	*	deny



50

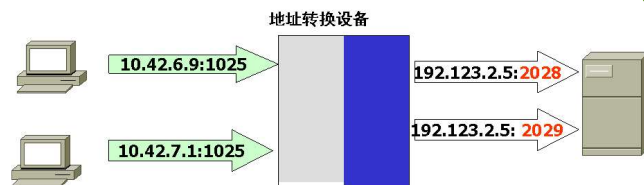
地址转换NAT

52

地址转换

@转换方式

- @静态地址转换，动态地址转换
- @静态地址转换+端口映射（Port Mapping）
- @动态地址转换+端口映射



54

包过滤技术的优缺点

@优点

- @可以保护所有的服务
- @对应用透明
- @相对于高层防火墙，较高的网络性能
- @保持Internet 端到端的特性
- @无状态，防火墙负载较轻

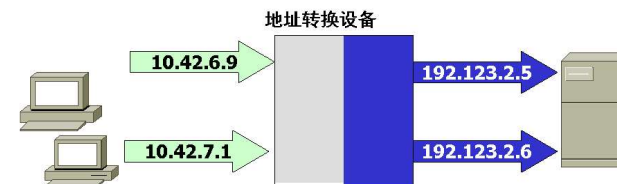
@缺点

- @没有认证功能
- @无法实施细粒度的访问控制政策
- @不能防范IP地址欺骗
- @不能防范IP分片攻击

51

地址转换NAT

- @类似路由器，工作在网络层。除了转发外，完成地址转换。
- @不能提供额外的安全性，但是可以隐蔽内部网络，节省地址空间。



53

地址转换

@优点

- @节省IP地址资源
- @隐蔽内部的网络

@缺点

- @地址转换破坏了IP包的完整性，破坏了Internet端到端的特性
- @动态地址转换必须保留状态，破坏了网络无状态的特性
- @Log 变得困难
- @端口映射使得包过滤变得困难

55

电路层网关 Circuit Gateway

56

电路层网关

@优点

- @支持所有TCP应用
- @保持状态，可以检测、防范SYN Flood类型的攻击
- @隐藏内部网络

@缺点

- @不支持UDP的应用
- @对应用不透明，应用软件必须经过socksified才能使用防火墙
- @保持状态，可能造成网络中断
- @性能的开销较大
- @防火墙本身易受DOS攻击

58

应用层代理

@应用代理型防火墙工作在OSI的最高层，即应用层；其特点是完全“阻隔”了网络流量，对每种应用服务编制特定的程序，对其实现监视和控制

- @可以支持身份认证功能
- @除了基于地址、协议、端口的控制以外，还可以支持应用层命令的过滤，例如FTP的get, put等命令。



60

电路层网关

@工作在传输层/会话层

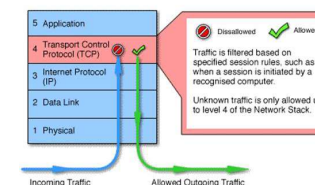
- @在客户和服务端之间建立了两个连接
- @对于收到的某个IP包，检查它是否属于某一个连接
- @不检查内容，不实现应用协议

@常见设备/软件

- @商业防火墙硬件/软件
- @免费软件：Socks

@相关标准

@rfc1928.txt

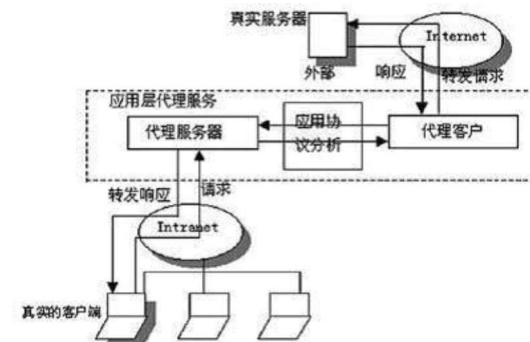


57

应用层代理

59

应用层代理



61

应用层代理

@第一代应用网关型防火墙(Application Gateway)

- @这类防火墙是通过一种代理(Proxy)技术参与到一个TCP连接的全过程
- @从内部发出的数据包经过这样的防火墙处理后,就好像是源于防火墙外部网卡一样,从而达到隐藏内部网结构的作用。
- @这种类型的防火墙被网络安全专家和媒体公认为是最安全的防火墙;它的核心技术就是代理服务器技术

62

应用层代理



@优点

- @功能强大: 用户认证、细粒度的访问控制
- @对于Web应用, 提供内容缓存功能(cache)

@缺点

- @协议相关, 需要对每一种应用协议编写Proxy程序
- @必须修改应用程序
- @通用计算机/软件实现, 影响网络性能

64

防火墙的配置结构

@单盒结构(Single-Box Architecture)

@屏蔽主机结构(Screened Host Architecture)

@屏蔽子网结构(Screened Subnet Architecture)

66

应用层代理

@第二代自适应代理型防火墙(Adaptive proxy)

- @近几年才得到广泛应用的一种新防火墙类型。它可以结合代理类型防火墙的安全性和包过滤防火墙的高速度等优点, 在毫不损失安全性的基础之上将代理型防火墙的性能提高10倍以上。
- @组成这种类型防火墙的基本要素有两个:
 - @自适应代理服务器(Adaptive Proxy Server)
 - @动态包过滤器(Dynamic Packet filter)。

63

防火墙的配置结构

65

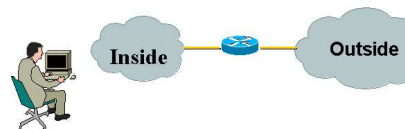
单盒结构

@屏蔽路由器: 简单的包过滤功能

- @优点: 投资小, 配置简单, 没有增加单一故障点
- @缺点: 在ACL较多时, 影响路由器的性能

@适用于如下环境

- @网络内部的主机安全性比较好, 规则简单
- @对性能、可靠性要求比较高



67

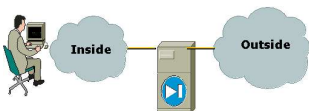
单盒结构

@双宿主机(Dual Home Host)

- @至少有两块网卡的通用计算机系统
- @可是包过滤软件/硬件、电路层网关、应用层代理
- @增加了单一故障点，影响网络吞吐量

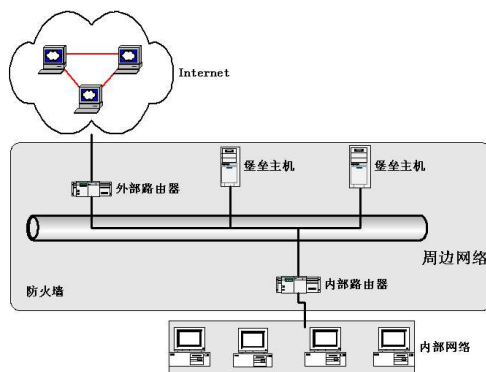
@适用于如下应用环境

- @去往Internet的流量比较小
- @对可靠性要求不高
- @不对外提供服务



68

屏蔽子网结构



70

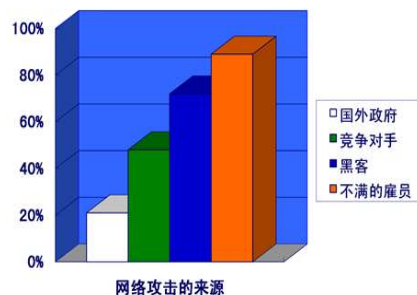
关于防火墙的争论



@防火墙破坏了Internet端到端的特性，阻碍了新应用的发展

@防火墙有先天性不足：没有解决网络内部的安全问题，防外不防内。

@防火墙给人一种误解，降低了人们对主机安全的意识



72

屏蔽主机结构

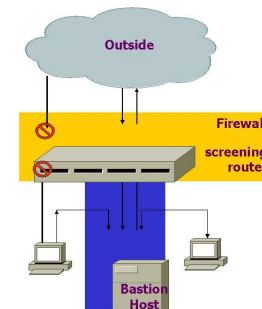
@屏蔽路由器

@堡垒主机

- @一台暴露在外部网络的攻击之下的计算机，要求安全性配置比较好的计算机。

@适用于

- @只对外提供较少的服务，外部的来的连接比较少
- @内部主机安全性配置较好



69

屏蔽子网结构

@DMZ(Demilitarized zone), 不设防区

- @通常放置DNS/Web/Email/FTP/Proxy Server等

@外部路由器

- @只允许对DMZ的访问
- @拒绝所有源地址为内部网络地址的包进入内网
- @拒绝所有不以内部网络地址为原地址的包离开网络

@内部路由器

- @保护内部网络，防止来自Internet或DMZ的访问
- @内部网络一般不对外提供服务，所以拒绝外部发起的一切连接，只允许内部对外的访问

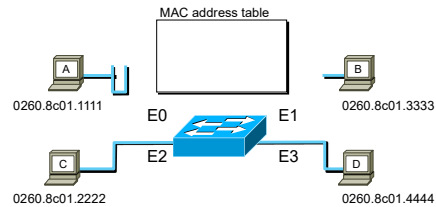
71

虚拟局域网技术VLAN

73

How Switches Learn Hosts Locations

@Initial MAC address table is empty



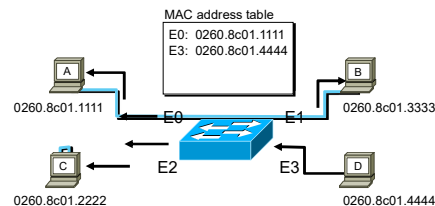
74

How Switches Learn Host Locations

@Station D sends a frame to station C

@Switch caches station D MAC address to port E3 by learning the source Address of data frames

@The frame from station D to station C is flooded out to all ports except port E3 (unknown unicasts are flooded)

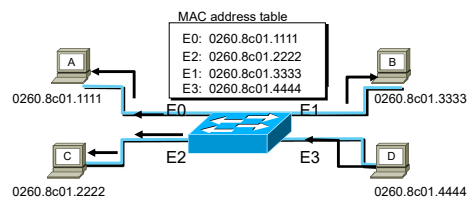


76

Broadcast and Multicast Frames

@Station D sends a broadcast or multicast frame

@Broadcast and multicast frames are flooded to all ports other than the originating port



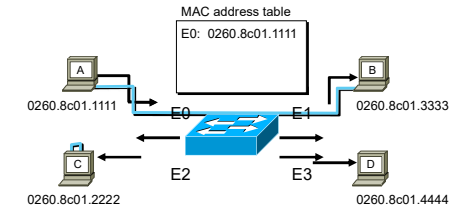
78

How Switches Learn Hosts Locations

@Station A sends a frame to Station C

@Switch caches station A MAC address to port E0 by learning the source address of data frames

@The frame from station A to station C is flooded out to all ports except port E0 (unknown unicasts are flooded)

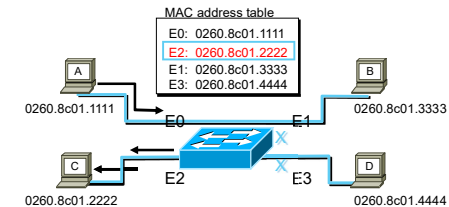


75

How Switches Filter Frames

@Station A sends a frame to station C

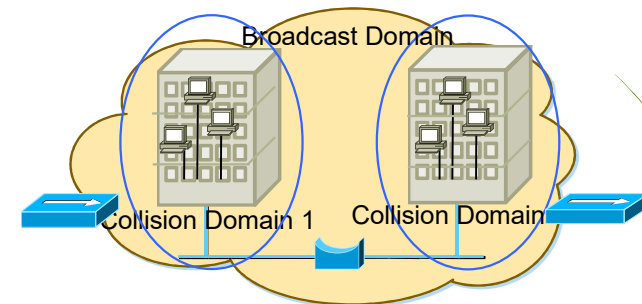
@Destination is known, frame is not flooded



77

冲突域和广播域

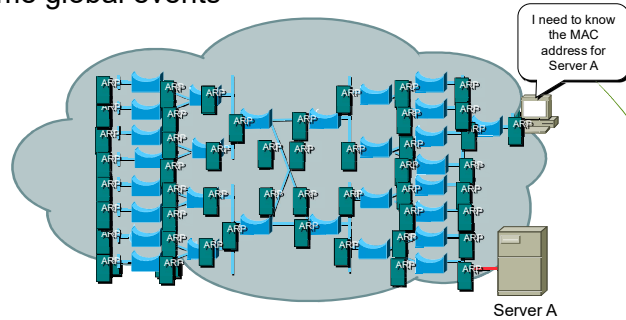
@Bridges terminate collision domains



79

广播风暴引起的性能问题

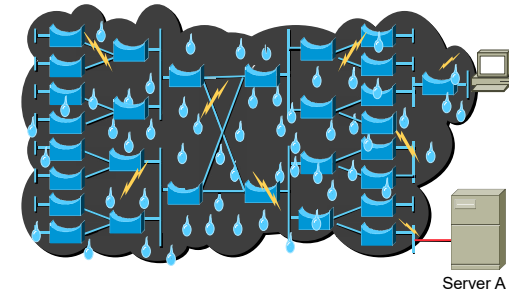
@Multicast, broadcast, and unknown destination events become global events



80

广播风暴

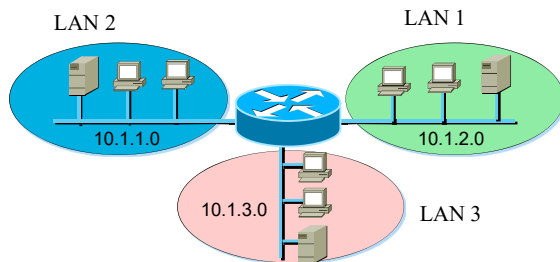
@Broadcasts can consume all available bandwidth
@Each device must decode the broadcast frame



81

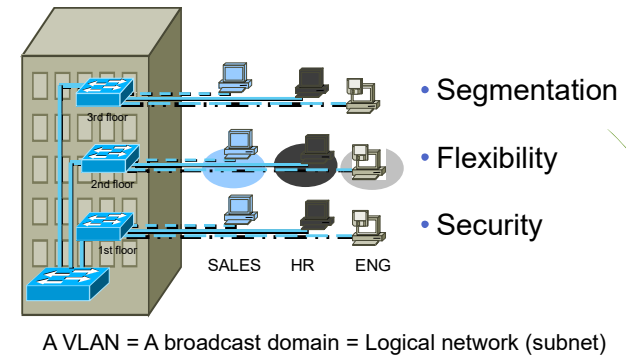
通过路由器隔离广播域

@LAN broadcasts terminate at the router interface



82

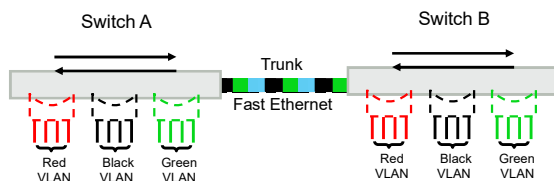
通过VLAN实现广播域的隔离



83

VLAN Operations

@Each logical VLAN is like a separate physical bridge
@VLANs can span across multiple switches
@Trunks carries traffic for multiple VLANs



84

VLAN 的类型

@基于物理端口划分的VLAN
@基于MAC地址划分的VLAN
@基于网络层划分的VLAN
 @基于协议的
 @基于子网地址的

85

基于物理端口划分的VLAN

@物理层VLAN(Port Based)

@把交换机端口分组

@优点

@配置简单

@缺点

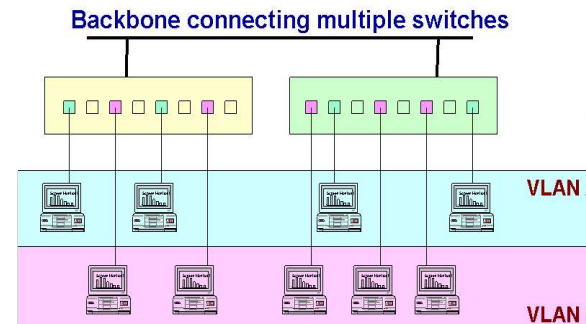
@不允许一个端口同时属于多过VLAN

@当终端计算机位置变化时，必须由管理员重新配置VLAN接口

Port	VLAN
1	1
2	1
3	2
4	1

86

基于物理端口划分的VLAN



87

基于MAC地址划分的VLAN

@二层VLAN(MAC Based)

@根据MAC Address分组，也称 “user-based VLAN”

@优点

@工作站物理移动时，不需要重新配置，依然属于原来的VLAN

@缺点

@在初始时所有的用户必须在至少一个VLAN上初始化

MAC Address	VLAN
1212354145121	1
2389234873743	2
3045834758445	2
5483573475843	1

88

基于MAC地址划分的VLAN



89

基于网络层划分的VLAN

@根据网络层协议类型划分的VLAN，比如基于IP、IPX等

Protocol	VLAN
IP	1
IPX	2

@根据网络层的子网分组（IP地址）划分的VLAN

IP Subnet	VLAN
166.111.1.0/24	1
166.111.2.0/24	2

@优点

@可以根据协议类型划分

@物理移动时，无需修改网络地址

@缺点

@交换机需要检查网络层协议信息，消耗资源

@不适用于没有层次结构的协议（不可路由）

90

VLAN的优缺点



@易于维护，容易解决人员位置的变动

@有效地控制广播流量，提高性能

@增强网络安全性

@互操作性问题

@标准滞后，实现上的不一致

@不同厂商实现方式不同，除非选择单一厂家的产品

@硬件设备、VLAN软件、管理软件

@设备的废弃会造成投资的浪费

@增加了管理的复杂性

91

课程主要内容

- @一、计算机网络安全概论
- @二、密码学基础
 - @密码学的基础知识
 - @对称密钥密码算法
 - @公开密钥密码算法
 - @密钥的分配
- @三、认证技术
 - @消息认证技术
 - @身份认证及其实例
- @四、网络安全协议
 - @网络层安全：IPsec、IKE
 - @传输层安全：SSL
 - @应用层安全：电子邮件安全、电子商务安全、WEB安全
- @五、各类专用技术
 - @访问控制技术：原理、实践

2

什么是访问控制列表ACL

ACL的分类

- @标准IP ACL(编号范围1至99以及1300至1999)限制条件为IP数据包头中的源IP地址。
- @扩展IP ACL(编号范围100至199以及2000至2699)限制条件为IP数据包头中的源、目的IP地址、协议类型和源、目的端口号。
- @其余编号范围的ACL是针对其它网络协议的。

Access List Type		Number Range/Identifier
IP	Standard	1-99 , 1300-1999
	Extended	100-199 , 2000-2699
	Named	Name (Cisco IOS 11.2 and later)
IPX	Standard	800-899
	Extended	900-999
	SAP filters	1000-1099
	Named	Name (Cisco IOS 11.2. F and later)

ACL实践

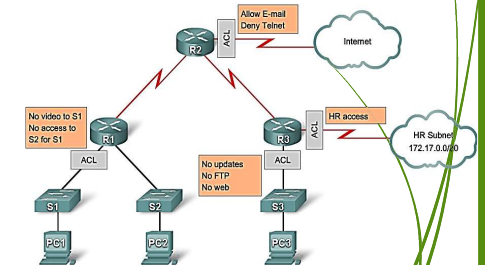
- @什么是访问控制列表(ACL)
- @ACL的执行规则
- @标准IP ACL的配置方法
- @扩展IP ACL的配置方法
- @ACL配置的核查与排错

什么是访问控制列表ACL?

@ACL(Access Control List)是一组预先定义好的规则，它被置于路由器的接口，根据进出接口的数据包头中的信息，控制该数据包能否穿越路由器

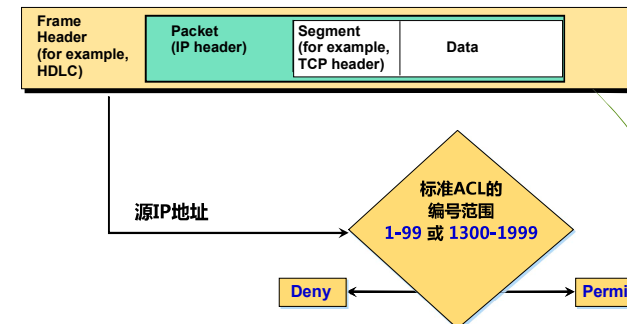
@ACL可执行下列任务：

- @限制网络流量增强网络性能
- @提供基本的安全访问控制，禁止某些特征的数据包访问而允许其它数据包通过
- @控制路由更新内容
- @区分特定的数据流类型(用于QoS、NAT等配置中)



标准ACL的检查内容

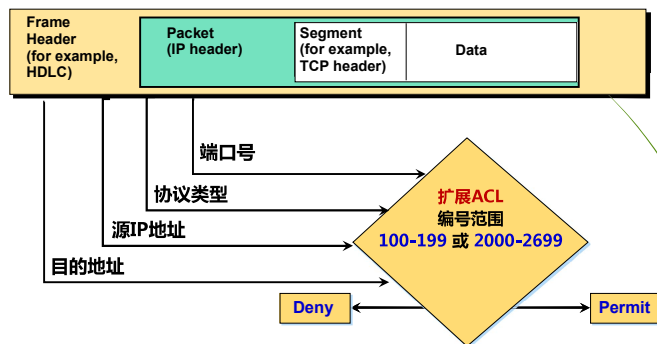
TCP/IP数据包的结构图



扩展ACL的检查内容

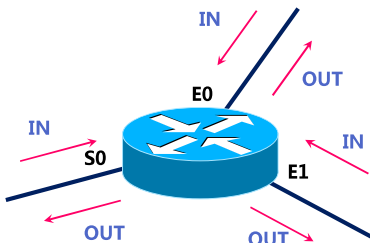


TCP/IP数据包的结构图

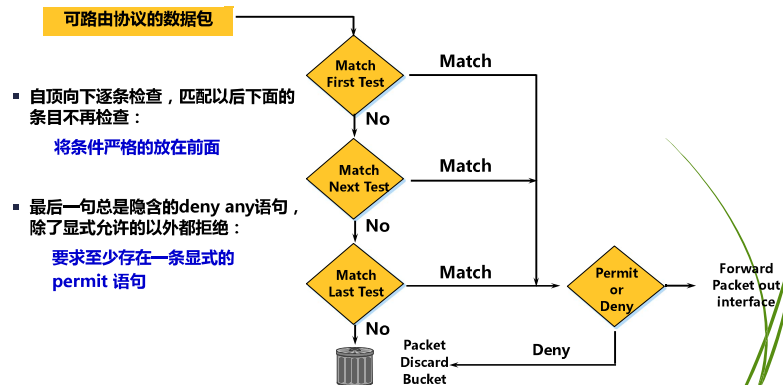


ACL的作用方向

@设置规则：对于每个协议，在每个路由器的接口的每个方向上只能设置一条ACL。

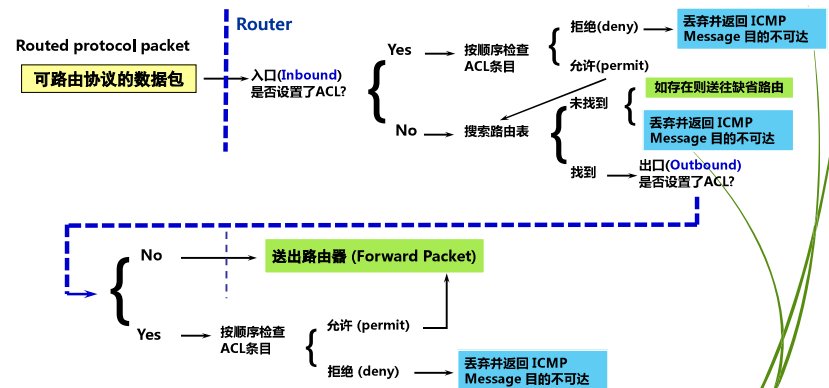


ACL的检查顺序



ACL的执行规则

ACL与路由的相互关系



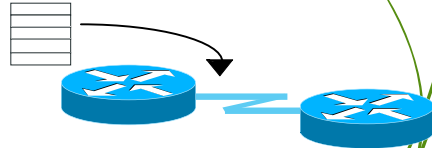
标准IP ACL的配置方法

ACL的配置步骤

@配置ACL的两个步骤:

1. 定制ACL的规则
2. 将规则应用到接口

```
access-list 1 deny 172.30.16.5 0.0.0.0
access-list 1 permit 172.30.16.0 0.0.0.255
access-list 1 deny 192.168.3.1 0.0.0.0
access-list 1 permit 192.168.1.0 0.0.0.255
access-list 1 deny any
```



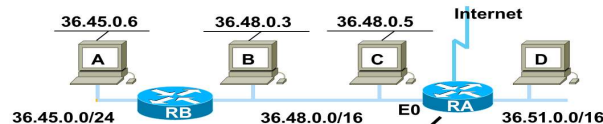
ACL通配符掩码的规则

Address	Wildcard	Matches
0.0.0.0	255.255.255.255	any address
131.108.0.0	0.0.255.255	network 131.108.0.0
131.104.7.11	0.0.0.0	host 131.104.7.11
255.255.255.255	0.0.0.0	local broadcast
131.111.8.0	0.0.7.255	subnet 131.111.8.0/21

@通配符掩码位是0表示必须匹配前面地址对应的比特

@通配符掩码位是1表示不必匹配前面地址对应的比特

标准IP ACL配置示例



```
RA(config)# access-list 1 deny 36.48.0.3 0.0.0.0
RA(config)# access-list 1 permit 36.48.0.0 0.0.255.255
! (Note: all other access implicitly denied)
RA(config)# interface ethernet 0
RA(config-if)# ip access-group 1 in
```

```
RA(config)# access-list 1 permit 36.48.0.0 0.0.255.255
RA(config)# access-list 1 deny 36.48.0.3 0.0.0.0 → 被前面的语句所包含, 因此无效
RA(config)# interface ethernet 0
RA(config-if)# ip access-group 1 in
```

标准IP ACL的配置命令

@定义标准ACL(编号范围 1- 99 和 1300 - 1999)

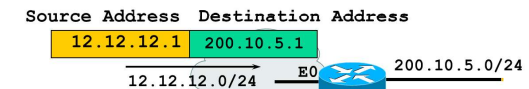
```
Router (config) #
access-list access-list-number { permit | deny }
{ source [ source-wildcard ] | any } [log]
```

@将ACL应用到特定接口

```
Router (config-if) #
ip access-group access-list-number { in | out }
```

@Log (可选参数) 将匹配该条目的数据包数量信息发送到控制台, 间隔5分钟发送一次

ACL的工作过程



```
access-list 3 permit 12.0.0.0 0.255.255.255
interface Ethernet 0
ip access-group 3 in
```

IP address	12.0.0.0	00001100	00000000	00000000	00000000
wildcard mask	0.255.255.255	00000000	11111111	11111111	11111111
	12.any.any.any	00001100	any	any	any
Example	16.12.12.1	00010000	00001100	00001100	00000001
	0.255.255.255	00000000	11111111	11111111	11111111

特殊的通配符掩码表示方法

@匹配所有的比特位 (match all)

172.30.16.29



Wildcard Mask: 0.0.0.0 (检查所有比特)

标准语法:

```
access-list 1 permit 172.30.16.29 0.0.0.0
```

可使用关键字 host 表示为:

```
access-list 1 permit host 172.30.16.29
```

如果不写通配符掩码隐含为 0.0.0.0, 例如:

```
access-list 1 permit 172.30.16.29
```

特殊的通配符掩码表示方法

@忽略所有的比特位 (ignore all)



Wildcard Mask: 255.255.255.255 (忽略所有比特)

标准语法:

```
access-list 1 permit 0.0.0.0 255.255.255.255
```

可使用关键字 any 表示为:

```
access-list 1 permit any
```

20

案例2：复杂通配符掩码的计算方法

@问题: 如果IP子网的变化范围改为 172.30.16.0/24 至 172.30.31.0/24, 该如何配置?

```
access-list 1 deny 172.30.31.0 0.0.0.255
access-list 1 permit 172.30.16.0 0.0.15.255
```

22

案例1：复杂通配符掩码的计算方法

@问题: 如果IP子网的变化范围172.30.16.0/24 至 172.30.31.0/24, 要求用一条语句表示

- 第3个字节前4个比特是相同部分, 必须完全匹配, 通配符为0;
- 后面4个比特是全排列, 为任意组合, 因此通配符为1。
- 通配符以十进制形式表示为: 0.0.15.255

172.30.16.0

00010000
00010001
00010010

...

00011111
通配符掩码 00001111

对应的十进制数
16
17
18
...
31
15

最终配置结果:

```
access-list 1 permit 172.30.16.0 0.0.15.255
```

21

案例3：复杂通配符掩码的计算方法

@问题: 如果IP子网允许通过的地址范围 200.10.5.10 ~ 15, 该如何配置?

分析最后一个字节的变化范围

00001010	10
00001011	11
00001100	12
00001101	13
00001110	14
00001111	15

通配符掩码 00000001 (1)

00001100 (12)
00001101 (13)
00001110 (14)
00001111 (15)

通配符掩码 00000011 (3)

最终配置结果:

```
access-list 3 permit 200.10.5.10 0.0.0.1
access-list 3 permit 200.10.5.12 0.0.0.3
```

23

扩展IP ACL的配置方法

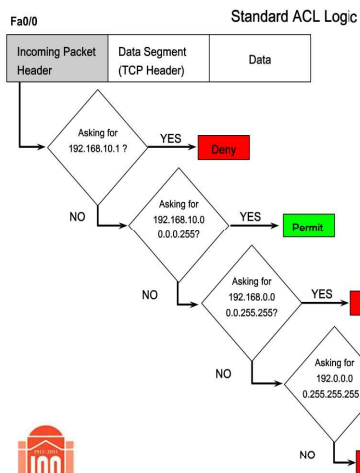
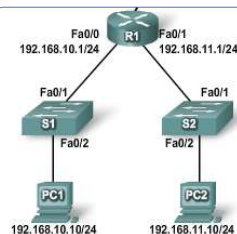
25

案例4：标准IP ACL配置

在R1的Fa0/0接口的入方向设置下列标准IP ACL:

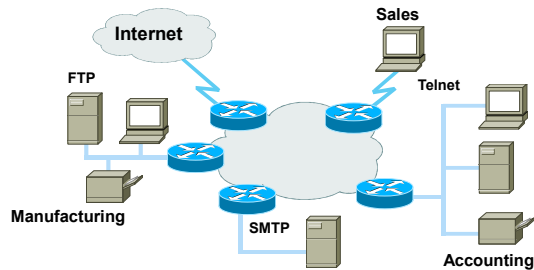
```
access-list 7 deny 192.168.10.10
access-list 7 permit 192.168.10.0 0.0.0.255
access-list 7 deny 192.168.0.0 0.0.255.255
access-list 7 permit 192.0.0.0 0.255.255.255
```

```
interface Fa0/0
ip access-group 7 in
```



标准IP ACL的局限性

@标准IP ACL只能根据包头中源IP地址控制流量



扩展IP ACL的配置命令

@定义扩展IP ACL(编号范围 100- 199 和 2000 - 2699)

```
Router (config) #
access-list access-list-number { permit | deny }
    { protocol | protocol-keyword }
    { source source-wildcard | any }
    { destination destination-wildcard | any }
    [ protocol-specific options ]
```

Protocol 字段关键字：ip, icmp, tcp, 和 udp 等。
Protocol-specific选项字段根据协议不同而变化。

@将ACL应用到特定接口

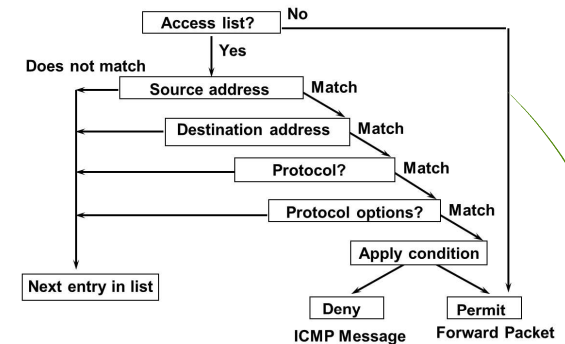
```
Router (config-if) #
ip access-group access-list-number { in | out }
```

TCP端口号对应的协议

@ 1	tcpmux TCP 端口服务多路复用	@ 53	domain 域名服务 (如 BIND)
5	rje 远程作业入口	63	whois++ WHOIS++, 被扩展了的 WHOIS 服务
7	echo Echo 服务	67	bootps 引导协议 (BOOTP) 服务;
9	discard 用于连接测试的空服务		还被动态主机配置协议 (DHCP) 服务使用
11	systat 用于列举连接的端口的系统状态	68	bootpc Bootstrap (BOOTP) 客户;
13	daytime 给请求主机发送日期和时间		还被动态主机配置协议 (DHCP) 客户使用
17	qotd 给连接了的主机发送每日格言	69	ttftp 小文件传输协议 (TFTP)
18	msh 消息发送协议	70	gopher Gopher 互联网文档搜寻和检索
19	chargen 字符生成服务; 发送无止境的字符流	71	netrjs-1 远程作业服务
20	ftp-data FTP 数据端口	72	netrjs-2 远程作业服务
21	ftp 文件传输协议 (FTP) 端口	73	netrjs-3 远程作业服务
22	ssh 安全 Shell (SSH) 服务	73	netrjs-4 远程作业服务
23	telnet Telnet 服务	79	finger 用于用户联系信息的 Finger 服务
25	smtp 简单邮件传输协议 (SMTP)	80	http 超文本传输协议 (HTTP)
37	time 时间协议	88	kerberos Kerberos 网络验证系统
39	rlp 资源定位协议	95	supdup Telnet 协议扩展
42	nameserver 互联网名称服务		
43	nicname WHOIS 目录服务		
49	tacacs 用于终端访问控制访问控制系统	@	端口号参见RFC 1700文档
50	re-mail-ck 远程邮件检查协议		

扩展IP ACL的优势

@扩展IP ACL可以根据数据源地址、目的地址、上层应用程序控制流量



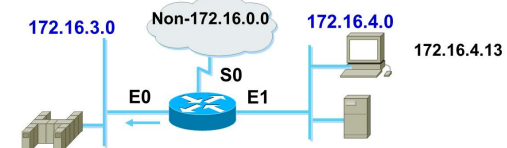
Protocol字段为TCP的语法

```
Router (config) #
access-list access-list-number { permit | deny } tcp
    { source source-wildcard | any }
    [ operator source-port | source-port ]
    { destination destination-wildcard | any }
    [ operator destination-port | destination-port ]
    [ established ]
```

Operator (可选) 其内容为: lt, gt, eq, neq 或 range .

案例：协议字段为TCP的扩展IP ACL

@问题：禁止来自子网172.16.4.0/24的流量访问目的子网172.16.3.0/24中FTP服务，其它流量允许通过。



```
access-list 101 deny tcp 172.16.4.0 0.0.0.255 172.16.3.0 0.0.0.255 eq 21
access-list 101 deny tcp 172.16.4.0 0.0.0.255 172.16.3.0 0.0.0.255 eq 20
access-list 101 permit ip any any
(implicit deny all)
(access-list 101 deny ip 0.0.0.0 255.255.255.255 0.0.0.0 255.255.255.255)

interface ethernet 0
ip access-group 101 out
```

Protocol字段为UDP的语法

Router (config) #

```
access-list access-list-number { permit | deny } udp
{ source source-wildcard | any }
[ operator source-port | source-port ]
{ destination destination-wildcard | any }
[ operator destination-port | destination-port ]
```

Operator (可选) 其内容为: lt, gt, eq, neq 或 range .

32

Protocol字段为ICMP的语法

Router (config) #

```
access-list access-list-number { permit | deny } icmp
{ source source-wildcard | any }
{ destination destination-wildcard | any }
[ icmp-type [ icmp-code ] | icmp-message ]
```

icmp-type 参见下一页内容

@ 部分ICMP 消息类型名称

@ 完整的ICMP信息参见
RFC 792文档

administratively-prohibited	information reply	port unreachable
alternate-address	mask-reply	reassembly-timeout
conversion-error	mask-request	redirect
dod-host-prohibited	mobile-redirect	router-advertisement
dod-net-prohibited	net-redirect	router-solicitation
echo	net-tos-redirect	source-quench
echo-reply	net-tos-unreachable	source-route-failed
general-parameter-problem	net-unreachable	time-exceeded

命名式ACL

@Cisco IOS 版本号11.2 及其后续版本支持命名式ACL 。

@命名式ACL的名字必须唯一且区分大小写。

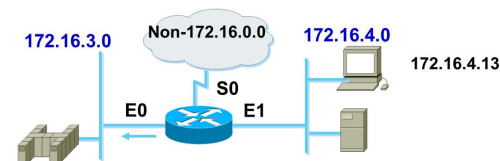
@命名式ACL的名字可以是纯字母或字母与数字的组合，也可以是纯数字形式(即编号式ACL属于命名式ACL的子集)

@编号式ACL不能单独删除一行语句，而命名式ACL能够单独删除一行。

36

案例：协议字段为UDP的扩展IP ACL

@问题：禁止来自子网172.16.4.0/24的流量访问172.16.3.0/24子网中的TFTP服务，其它流量允许通过。



```
access-list 101 deny udp 172.16.4.0 0.0.0.255 any eq 69
access-list 101 permit ip any any
(implicit deny all)
interface ethernet 0
ip access-group 101 out
```

33

案例：协议字段为ICMP的扩展IP ACL

@问题：要求在RB上能PING通RA的S0/1接口地址，但不允许来自RA及互联网上的流量PING RB的任何接口地址，其它流量允许通过，应该如何配置？

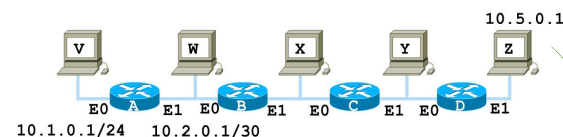


```
RB(config)# access-list 100 deny icmp any host 192.168.1.1
RB(config)# access-list 100 deny icmp any host 36.45.0.1
RB(config)# access-list 100 permit ip any any
RB(config)# interface S0/0
RB(config-if)# ip access-group 100 in
```

35

案例：ACL

@问题：在路由器A右侧的网络中，只允许主机 Z Telnet路由器A，其它类型流量不做限制，要求使用扩展IP ACL，如何设置？



```
RA(config)# access-list 100 permit tcp host 10.5.0.1 host 10.2.0.1 eq 23
RA(config)# access-list 100 permit tcp host 10.5.0.1 host 10.1.0.1 eq 23
RA(config)# access-list 100 deny tcp any host 10.2.0.1 eq 23
RA(config)# access-list 100 deny tcp any host 10.1.0.1 eq 23
RA(config)# access-list 100 permit ip any any
RA(config)# interface Ethernet1
RA(config-if)# ip access-group 100 in
```

37

ACL配置的核查与排错

38

显示访问控制列表的命令

@显示所有协议的访问控制列表

```
Router #  
show access-lists [ access-list-number ]
```

@显示IP协议的访问控制列表

```
Router #  
show ip access-lists [ access-list-number ]
```

@案例

```
Router# show ip access-lists  
Extended IP access list 101  
deny udp any any eq ntp  
permit tcp any any  
permit udp any any eq tftp  
permit icmp any any  
permit udp any any eq dns  
Router#
```

40

检查接口上是否设置了ACL

```
Router# show ip interface e0  
Ethernet0 is up, line protocol is up  
Internet address is 10.1.1.11/24  
Broadcast address is 255.255.255.255  
Address determined by setup command  
MTU is 1500 bytes  
Helper address is not set  
Directed broadcast forwarding is disabled  
Outgoing access list is not set  
Inbound access list is 1  
Proxy ARP is enabled  
Security level is default  
Split horizon is enabled  
ICMP redirects are always sent  
ICMP unreachable are always sent  
ICMP mask replies are never sent  
IP fast switching is enabled  
IP fast switching on the same interface is disabled  
IP Feature Fast switching turbo vector  
IP multicast fast switching is enabled  
IP multicast distributed fast switching is disabled  
<text omitted>
```

ACL的配置规则总结



- @对于每个协议，在每个路由器接口的每个方向上只能设置一条ACL，后写入的覆盖先前的
- @自顶向下逐条检查，匹配后其余条目不再检查：将条件严格的放在前面
- @最后一句是隐含的deny any语句，因此至少存在一条显式的permit语句
- @新增加的语句总是置于最后一行
- @未定义的ACL (空ACL)相当于permit any
- @访问控制列表不能限制起源于本路由器的流量

课程主要内容

- @一、计算机网络安全概论
- @二、密码学基础
 - @密码学的基础知识
 - @对称密钥密码算法
 - @公开密钥密码算法
 - @密钥的分配
- @三、认证技术
 - @消息认证技术
 - @身份认证及其实例
- @四、网络安全协议
 - @网络层安全：IPsec、IKE
 - @传输层安全：SSL
 - @应用层安全：电子邮件安全、电子商务安全、WEB安全
- @五、各类专用技术
 - @访问控制技术：原理、实践
 - @**入侵检测技术：**
用户入侵、软件入侵

2

主要内容

- @**用户入侵**
 - @**入侵者**
 - @**入侵技术**
 - @**入侵检测**
 - @审计方法
 - @基于统计的异常检测
 - @阈值检测、轮廓检测
 - @基于规则的入侵检测
 - @异常检测、渗透鉴别
 - @蜜罐技术
- @**软件入侵**
 - @恶意软件
 - @陷门
 - @逻辑炸弹
 - @特洛伊木马
 - @Zombie
 - @病毒
 - @蠕虫

4

入侵者

- @入侵者通常指黑客和解密高手，一般分成：
 - @假冒者
 - @未经授权使用计算机的人和穿透系统的存取控制冒用合法用户帐号的人。
 - @非法者
 - @未经授权访问数据、程序和资源的合法用户，或者已经获得授权访问，但是错误使用权限的合法用户。
 - @秘密用户
 - @夺取系统超级控制并使用这种控制权逃避审计和访问控制或者抑制审计记录的个人。

6

入侵技术

- @网络系统面临的一个严重的安全问题是用户或者软件的恶意入侵。
 - @用户入侵包括未授权登录，或授权用户非法取得更高级别的权限和操作。
 - @软件入侵方式包括病毒、蠕虫和特洛伊木马等。
- @入侵技术是网络安全和系统安全的交叉领域。
- @用户入侵和软件入侵都与网络安全有关，因为攻击大部分都是通过网络实现的，但又不仅仅是基于网络的攻击。
 - @终端用户入侵本地系统可以不通过网络。
 - @病毒或特洛伊木马可以通过磁盘传播到系统。
 - @蠕虫才是网络特有的入侵方式。

3

用户入侵：入侵者

5

入侵者



- @假冒者一般是外部使用者，非法者一般是内部人员，秘密用户内外部用户均有可能。
- @入侵者的攻击可能是友善的，也可能是险恶的。
- @善意的入侵者只想探索一下网络是什么，尽管占用了资源、降低了合法使用者的系统性能，但还是可以容忍的。
- @但是由于无法预知入侵者是善意的，还是恶意的，所以要严格控制入侵问题。

7

用户入侵：入侵技术

8

入侵技术

@常用的获取口令的技术有：

- @使用系统提供的标准帐户和默认口令。
 - @很多管理员不愿意改变默认值。
- @穷尽所有的短口令。
- @尝试系统在线词典中的单词或类口令单词列表。
- @收集用户的信息。
- @尝试用户的电话号码、房间号码、身份证号码等。
- @尝试本国所有合法的牌照号码。
- @使用特洛伊木马逃避访问控制。
- @窃听远程用户和主机系统之间的线路。

10

入侵技术



@对于入侵者防范的主要措施有：检测和阻止。

@检测注重掌握攻击成功前后所采用的思路。

@阻止在于挫败所有可能的攻击。

12

入侵技术

@入侵者的动机是获取系统访问权或者扩大在系统中的权限范围。

@入侵者需要获得系统保护信息，这些信息一般以用户口令的形式存在。

- @系统通常保存一份授权用户和授权用户口令的文件。
- @如果这个文件没有采取保护措施，攻击者很容易访问文件，获取口令。进而利用口令，以合法用户的身份登录系统，进行破坏活动。

9

入侵技术

@口令文件可以采用如下方式进行保护：

- @单向加密
 - @系统只保存用户口令的密文。当用户要求验证口令时，系统加密此口令，与保存的密文对比，判断真假。
- @访问控制
 - @对口令文件的访问仅限于几个帐号。
- @使用上述任何一种口令文件保护方法，入侵者都很难获取口令。

11

用户入侵：入侵检测

13

入侵检测

- @不存在最好的入侵阻止系统，入侵检测一直是研究热点。
 - @检测到入侵的速度足够快：在迫害发生或者危及数据之前，识别出入侵者，并将其驱逐出去。
 - @即使没有非常及时地检测到入侵者，入侵检测速度越快，破坏程度会越低，恢复地越快。
 - @有效的入侵检测系统可以看成是阻止入侵的有效屏障。入侵检测可以收集入侵技术的信息，用以增强入侵阻止工具。
- @入侵检测建立的前提是假设入侵者的行为在某些情况下不同于合法用户的行为。
- @当然，入侵的攻击和合法用户正常使用资源的区别不可能十分明显，甚至，他们的行为含有很多相似之处。

14

审计记录

- @每个审计记录可以包含如下几个域：
 - @主体：行为的发起者。主体可以分为不同的访问类别，类别之间可以重叠。
 - @动作：主体对一个对象的操作或者联合一个对象完成的操作。
 - @客体：行为的接收者。
 - @异常条件：若返回有异常，则标识出该异常情况。
 - @资源使用：大量元素的列表。每个元素都给出某些资源使用的数量。
 - @时间戳：动作发生时用来标识时间的记录。

16

统计异常行为的入侵检测

- @统计异常行为的入侵检测：收集一段时间内合法用户的行为，用统计测试来观测其行为，判定该行为是否是合法用户行为。
 - @阈值检测：这种方法根据各种事件发生的频率确定阈值，阈值的确定与用户无关。
 - @基于轮廓的检测：为每个用户建立活动轮廓，用于检测每个帐号行为的变化。
- @统计方法能够有效地对付假冒者，因为假冒者不可能模仿他们盗用帐号的行为模式，但统计方法不能防范合法者。
 - @对于合法者的攻击，基于规则的方法可以识别攻击事件和顺序，从而发现入侵。

18

审计记录

- @入侵检测的基本工具是审计记录。
 - @用户活动记录应该作为入侵检测系统的输入。
- @审计记录有两种方法：
 - @原始审计记录
 - @几乎所有的多用户操作系统都有收集用户活动信息的审计软件。使用这些信息的好处是不需要额外的软件开销，缺点是审计记录可能没有包含所需要的信息，或者信息没有以方便的形式存放。
 - @检测专用的审计记录
 - @审计软件只收集入侵检测系统需要的信息。需要额外的软件开销。

15

入侵检测方法

- @常用的入侵检测方法包括：
 - 基于统计的方法和基于规则的方法。
- @基于统计的方法试图定义正常的、期望的行为，基于规则的方法定义正确的行为。
 - @阈值检测、基于轮廓的检测
- @基于规则的入侵检测定义一个规则集，用于判定给定行为是否为入侵者行为。
 - @异常检测、渗透鉴别
- @事实上，很多系统将两种方法结合起来对付更大范围的攻击。

17

统计异常行为的入侵检测

- @阈值检测与一段时间间隔内的特殊时间发生的次数有关，如果发生次数超过了期望的合理次数，则认为可能存在入侵。
- @阈值分析本身是粗糙而效率不高的检测器，因为用户不断变化，阈值检测可能出错，只有和其它高级技术共同使用，阈值检测才会有效。

19

统计异常行为的入侵检测

- @基于轮廓的异常检测归纳出每个用户过去的行为特征或者用户组过去行为的特征，用于发现有重大偏差的行为。
 - @轮廓一般包括多个参数集，所以只发生一个参数的偏差不足以产生一个警告。
- @基于轮廓的异常检测的最大优点是不用先知安全漏洞知识。
 - @检测程序首先学习什么是正常行为，然后去查找行为的偏差。与系统特征和脆弱性无关。可广泛用于多种系统。

20

基于规则的入侵检测

- @基于规则的入侵检测系统是通过观察系统中的事件，运用规则集判定一个给定的活动模式是否可疑。
- @基于规则的渗透鉴别建立在专家系统之上，关键在于使用规则去鉴别已知的渗透或利用已知弱点的渗透。
 - @规则不是由分析审计记录得到的，而是专家系统定义的，可用于识别可疑行为。
- @基于规则的入侵检测分成两种类型：
 - @异常检测：定义规则，用于检测当前行为和以前使用的模式的差别。
 - @渗透鉴别：用专家系统来查找可以的行为。

22

蜜罐技术

- @入侵检测技术中相对较新的创新是蜜罐。蜜罐是诱导潜在攻击者远离重要系统的一个圈套。
- @蜜罐的功能是：
 - @转移攻击重要系统的攻击者。
 - @收集攻击者活动的信息。
 - @希望攻击者在系统中逗留足够长的时间，使管理员能对此攻击做出响应。
- @蜜罐系统使用的工具包括灵敏的监视器和事件日志。
 - @事件日志用以检测访问和收集攻击者活动的信息。

24

统计异常行为的入侵检测

- @基于轮廓入侵检测的度量机制如下：
 - @计数器、标准值
 - @间隔计时器、资源利用
- @上述度量机制可以和下面一些方法配合使用：
 - @均值和标准偏差
 - @多变量模型
 - @马尔可夫处理
 - @时间序列模型
 - @操作模型

21

基于规则的入侵检测

- @基于规则的异常检测使用基于规则的方法分析审计记录，确定使用模式，并自动产生描述此模式的规则。
 - @规则是标识用户、程序、特权、时间槽、终端等过去行为的模式。通过观察当前行为，将每个行为与匹配规则集进行匹配，判定它是否符合某个历史行为模式。
 - @此方法和强度上类似基于统计的异常检测。而且与基于统计的异常检测一样，基于规则的异常检测无需知道系统内部的安全脆弱性。

23

蜜罐技术



- @任何对蜜罐的攻击，系统都会给出攻击成功的假相。所以系统管理员可以在不暴露真正在工作的系统条件下，有时间转移、记录、跟踪攻击者。
- @蜜罐系统充满合法用户无法访问、但表面看起来有价值的虚假信息。因此，任何对蜜罐的访问都是可疑的。
- @最初是使用一台有IP地址的计算机作为蜜罐引诱黑客。最近的研究集中在建立整个蜜罐网络，用来模拟一个企业。

25

软件入侵

病毒、蠕虫和木马

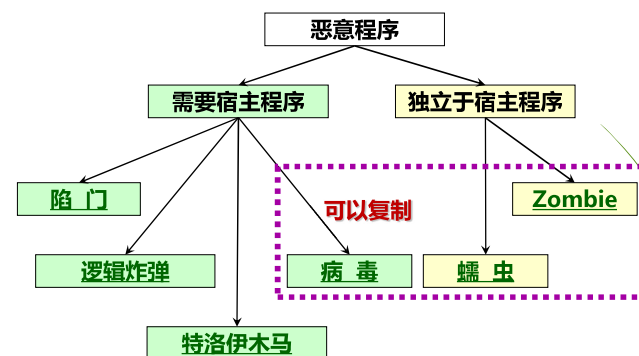
特性	病毒	蠕虫	木马
宿主	需要	不需要	需要
表现形式	不以文件形式存在	独立的文件	伪装成其他文件
传播方式	依赖宿主文件或介质	自主传播	依靠用户主动传播
主要危害	破坏数据完整性、系统完整性	侵占资源	留下后门窃取信息
传播速度	快	极快	慢

恶意软件：陷门

恶意软件

- @软件入侵就是一些恶意软件，利用计算机系统的弱点进行攻击的恶意程序。
- @这些恶意程序可以分成两类：依赖于宿主程序的和独立于宿主程序的威胁。也可以按照是否进行复制进行区分。
 - @依赖于宿主程序的恶意软件是在宿主程序被调用来执行某一特定功能时被计划的程序段。
 - @独立于宿主程序的恶意软件是一个程序段（病毒）或者一个独立的程序（蠕虫、细菌），当它执行的时候，可能会对自身进行复制，这些复制品将会在本系统或者其它系统中被激活。

恶意软件



陷门（后门）

- @陷门是一个程序的秘密入口，通过它用户可以不按照通常的访问步骤获得访问权。
- @许多年来，程序员合理地利用陷门进行程序调试和测试。当陷门被一些无耻之徒用来作为获得未授权的访问权的手段时，陷门就成为一种威胁。
- @要实现对陷门的控制很困难，解决办法是程序的开发和软件的更新。

陷门的分类

@从功能上，陷门可以分成三个级别：

- @系统级：在系统层能访问数据和进程
- @应用级：逃逸安全机制的合法程序
- @密码级：能够以某种方式看懂密文

@从控制方法上，陷门分成以下几类：

- @本地权限的提升
 - @对系统有访问权的攻击者变换其权限等级成为管理员，然后攻击者可以重新设置该系统或访问人和存储在系统中的文件。

32

陷门的分类（续）

@从控制方法上，陷门分成以下几类：

- @单个命令的远程执行
 - @攻击者可向目标计算机发送消息。每次执行一个单独的命令，陷门执行攻击者的命令并将输出返回给攻击者。
- @远程命令行解释器访问
 - @如远程Shell，这类陷门允许攻击者通过网络快速直接地键入受害计算机的命令提示。其比“单个命令的远程执行”要强大得多。
- @远程控制GUI
 - @攻击者可以看到目标计算机的GUI，控制鼠标的移动，输入对键盘的操作，这些都通过网络实现。

33

后门的安装和案例



@自己植入

- @物理接触或入侵之后

@漏洞

- @病毒
- @蠕虫
- @恶意移动代码

@社会工程（欺骗受害者自己安装）

- @Email
- @远程共享
- @BT下载
- @.....

@著名的案例

- @2001年，Borland的数据库软件Interbase发现隐藏了七年的后门，程序中包含一个万能用户名和密码：
 - @ ID: politically
 - @ PW: correct
- @用户包括波音、诺基亚、摩托罗拉和波士顿股票交易所、Sun等

34

恶意软件：逻辑炸弹

35

逻辑炸弹



@逻辑炸弹是最早出现的恶意软件之一，它预先规定了病毒和蠕虫的发作时间。逻辑炸弹是嵌在合法程序中的、只有当特定的事件出现时才会进行破坏的一组程序代码。

@逻辑炸弹的典型例子是Tim Lloyd事件。

- @Tim Lloyd设计的逻辑炸弹是他所在公司蒙受了1000万美元的损失，并导致80位员工失业。最后，Tim Lloyd被处以41个月的监禁和200万美元的赔偿金。

36

恶意软件：特洛伊木马

37

特洛伊木马

- @特洛伊木马程序是一种实际上或者表面上看起来有某种有用功能的程序，它内部含有隐蔽代码，当被调用时会产生一些意想不到的后果。
 - @欺骗用户或者系统管理员安装
 - @在计算机上与“正常”的程序一起混合运行，将自己伪装得看起来属于该系统。
- @如果一个程序仅提供远程访问，那么它只是一个后门。如果攻击者将这些后门功能伪装成某些其他良性程序，那么就涉及到真正的特洛伊木马。

38

木马软件的构成

- @木马系统软件一般由木马配置程序、控制端程序和木马程序(服务器程序)等三部分组成。
- @木马程序
 - @也称服务器程序，它驻留在受害者的系统中，非法获取其操作权限，负责接收控制端指令，并根据指令或配置发送数据给控制端。
- @木马配置程序
 - @设置木马程序的端口号、触发条件、木马名称等，使其在服务端藏得更隐蔽，有时该功能被集成在控制端程序菜单内，不单独作为一个程序。
- @控制端程序
 - @控制远程服务器，有些集成了木马配置的功能。

40

木马软件的常用手段



- @名字欺骗：修改、模仿文件名以欺骗用户
 - @Beauty.jpg.exe
 - @httpd,iexplore,notepad,ups,svchost...
- @文件捆绑：恶意程序与正常程序捆绑
- @软件下载，软件本身带毒
- @Html传播
 - @网页病毒的传播方式
 - @Flash传播网页、email传播网页
 - @Exe2bmp
 - @正常网页中携带

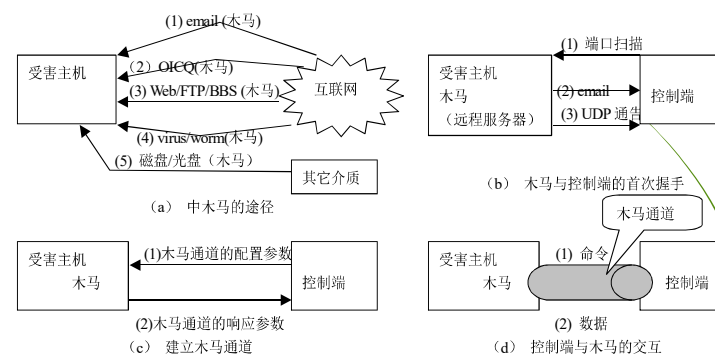
42

特洛伊木马

- @特洛伊木马使潜伏者执行非授权的功能。
 - @例如：利用已经修改的编译程序向正在被编译的程序中插入一些附加代码，这样的特洛伊木马就很难检测出来。
 - @例如：特洛伊木马对数据破坏，看起来在执行某个功能，实际在删除用户的文件，从而造成难以恢复的破坏。

39

木马的工作原理



41

恶意软件：Zombie

43

Zombie



@Zombie秘密地接管其它依附在Internet上的计算机，并使该计算机发动攻击，而且这种攻击很难通过追踪Zombie的创建者查出来。

@Zombie被用在对拒绝服务的攻击上，尤其是对Web网站的攻击。它被放置在成百上千的、属于第三方的计算机中，通过向Internet发送不可抵抗的攻击取得对目标Web网站的控制。

44

病毒

@病毒是一种可以通过修改自身来感染其它程序的程序，这种修改包括对病毒程序的复制。复制后生成的新病毒同样具有感染其他程序的功能。

@通过寄居在宿主程序上，计算机病毒可以暂时控制该计算机的操作系统盘。

@没有感染的软件一旦在带毒计算机上运行，就会在新程序中产生病毒拷贝。病毒就会通过软盘或者网络传播开来。

46

病毒的破坏程度

@病毒发作体现出来的破坏程度是不同的，有些是无害的，例如屏幕上的一些干扰信息；有些会带来巨大的危害，例如破坏程序或者数据。

@多数病毒都是基于某种特定的方式进行工作的，例如某个操作系统或者某个特定的硬件平台。因此，攻击者经常利用系统的细节和弱点来设计病毒程序。

48

恶意软件：病毒

病毒原理、引导性病毒

病毒的生命周期

@潜伏阶段：

@这个阶段病毒处于休眠状态，这些病毒最终会被某些条件激活。并不是所有的病毒都会经历此阶段。

@传染阶段：

@病毒程序将自身复制到其他程序或者磁盘的某个区域上，每个被感染的程序又因此包含了病毒的复制品，从而也就进入了传染阶段。

@触发阶段：

@病毒在被激活后，会执行某个特定的功能从而达到某种既定的目的。

@发作阶段：

@病毒在触发条件成熟的时候，即可在系统中发作。

47

病毒的种类

@寄生性病毒

@将自身依附在可执行文件上并对自身进行复制。当受感染文件被执行后，它又会继续寻找其他的可执行文件并对其进行感染。

@引导扇区病毒

@此类病毒感染主引导记录或者引导记录，在系统从含有病毒的磁盘上引导装入程序时进行传播。

@常驻存储器病毒

@该病毒以常驻系统的程序的形式寄居在主存储器上面，它会感染所有的文件。

@隐蔽性病毒

@此病毒目的是为了躲避反病毒软件的检测。

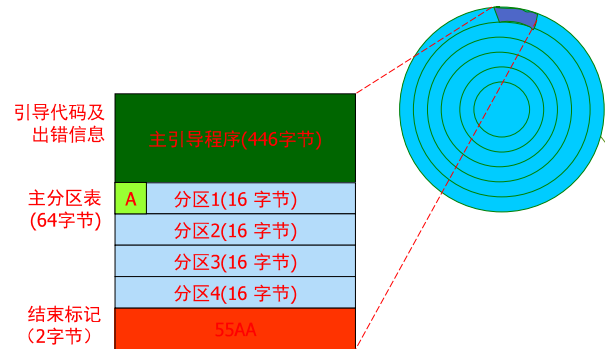
@多态性病毒

@这种病毒在每次感染时，放入宿主程序的代码互不相同，不断变化。

49

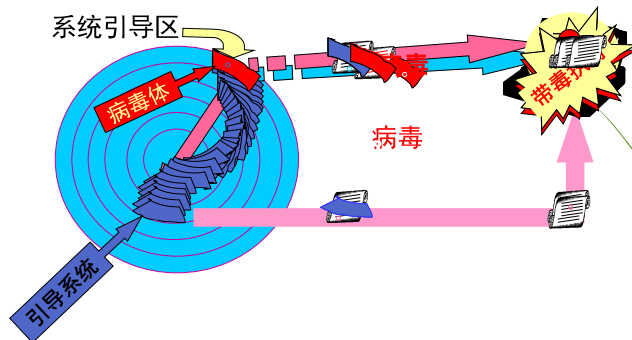
引导型病毒：引导记录

@主引导记录 (MBR)



50

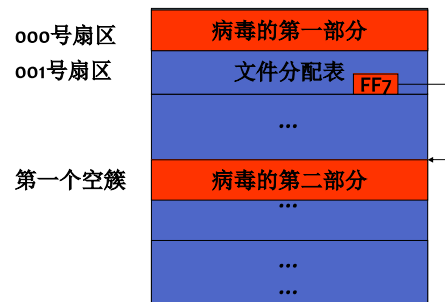
引导型病毒：感染与执行过程



52

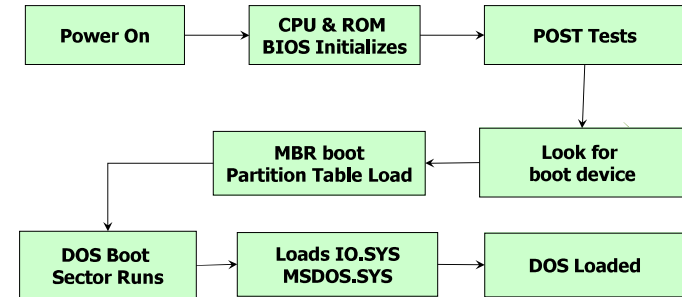
引导型病毒举例

@小球病毒(Bouncing Ball)：在磁盘上的存储位置



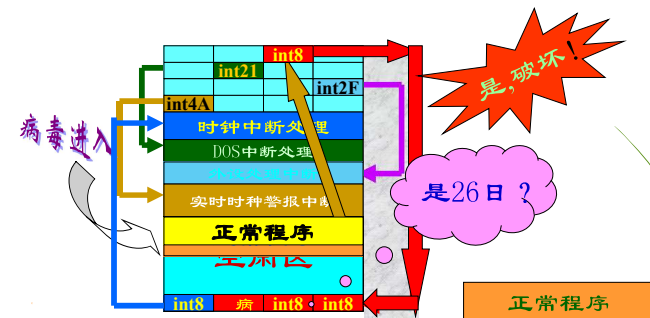
54

引导型病毒：系统引导过程(DOS)



51

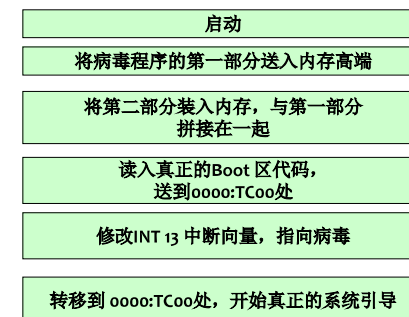
病毒的激活过程



53

引导型病毒举例

@感染后的系统启动过程



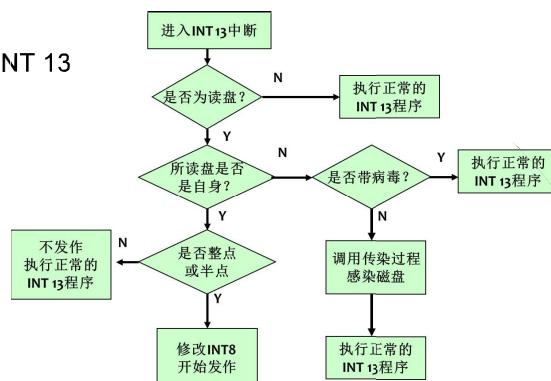
55

引导型病毒举例



@触发条件

@修改后的INT 13



56

什么是蠕虫Worm

@美国传统词典

@A program that replicates itself and interferes with software function or destroys stored information.



@一种能够自我复制的计算机程序，它利用网络上计算机系统的漏洞自主的将自己复制到其它计算机上。

- @一个独立的计算机程序，不需要宿主。
- @自我复制，自主传播。
- @占用系统或网络资源、破坏其他程序。
- @不伪装成其他程序，靠自主传播。

58

蠕虫的危害

@蠕虫的传播造成网络拥塞。

@对计算机系统和信息安全造成破坏。

@蠕虫的破坏力巨大

- @CodeRed蠕虫在爆发后的9小时内就攻击了25万台计算机，造成的损失估计超过20亿美元。
- @Nimda造成的损失评估数据从5亿美元攀升到26亿美元后，继续攀升，到现在已无法估计。

60

恶意软件：蠕虫

蠕虫原理、Morris 蠕虫、CodeRed蠕虫

57

蠕虫病毒

@电子邮件病毒将自身从一台计算机传播到另外一台计算机，因此具有蠕虫的一些特征。但它仍需要人为地手工移动。

@蠕虫病毒会时刻寻找更多的计算机并伺机传染。那些被感染的机器又会作为一种自动发射缓冲器，以进一步感染其它机器。

@只要系统中的蠕虫处于活动状态，它就对系统做出巨大的破坏行为。

59

网络蠕虫病毒

@网络蠕虫病毒利用以下网络工具进行传播：

@电子邮件设备

@蠕虫病毒将自身的拷贝以邮件的形式发送到其它系统。

@远程执行功能

@蠕虫病毒在其它系统执行自身的拷贝。

@远程登录功能

@蠕虫病毒以用户身份进入远程系统，然后使用命令将自身从一台计算机复制到另外一台计算机。

61

蠕虫的历史回顾

- @1980年 Xerox PRAC
- @1988年11月2日 Morris Worm
- @1989年10月16日 WANK Worm
- @1998年5月 ADM Worm
- @1999年9月 Millennium
- @2001年1月 Ramen Worm
- @2001年3月23日 Lion Worm
- @2001年4月3日 Adore Worm

62

蠕虫的历史回顾

- @2003年 冲击波蠕虫和“Blast清除者”蠕虫
- @2003年 口令蠕虫
- @2003年 SQL SLAMMER蠕虫
- @2004年 震荡波蠕虫
- @2005年 黛蛇蠕虫
- @2006年 “魔波”蠕虫
- @2007年 “Nimaya(熊猫烧香)”病毒
- @2008年 “机器狗”病毒
- @2009年 “飞客”蠕虫

64

Morris Worm的诞生

- @调查很快展开，结果更出人意外：是美国国家安全局一位专家的儿子Robert Morris干的！他还只是康奈尔大学一个二十出头的研究生。
- @Robert Morris 编的这个程序叫Worm，只有99行。严格地说它不是病毒，因为病毒需要一个宿主程序。而Worm程序则是自我复制，自行传播，更具危害性。

66

蠕虫的历史回顾

- @2001年5月 Cheese Worm
- @2001年5月 Sadmin/IIS Worm
- @2001年7月19日 CodeRed Worm
 - @催生国家级网络安全应急组织
- @2001年9月18日 Nimda Worm
- @2002年9月14日 Slapper Worm

63

Morris Worm的诞生

- @1988年11月3日，当数千名计算机系统管理员像往日一样轻松自在地赶到工作岗位时，他们惊呆了，系统瘫痪！原因不明！
 - @当他们知道类似系统瘫痪的机器有6000多台时，没有人不怀疑这是一次阴谋，互联网络受到了攻击，十分之一的机器完全瘫痪，其它系统也是缓慢如牛。
- @在人们正对互连网络充满憧憬之时，这样一次事件无异于迎面兜盆冷水，让人充满恐惧。

65

Morris Worm的诞生

- @刚开始，Robert Morris并不是第一嫌疑犯。因为他是一个好学生，没有前科；更重要的一点是他的父亲曾经对Internet的设计有过意义深远的影响。
- @Robert Morris很快发现Worm程序以一种比他的预计快得多的速度正在进行自我复制和感染网络中的机器。原因是：程序中有缺陷。
- @当认识到事情严重性时，他与在哈佛的一个朋友联系并讨论解决的办法。最后，他们从哈佛向整个网络发了一封匿名信，指导程序员们如何杀死Worm程序以及如何防止被感染。

67

Morris Worm的诞生

@Robert Morris为什么编制这个程序？

@Robert Morris说原本只想通过自己编写的程序，估计当时的互联网大小。他在程序中设计了计数功能，可以控制“蠕虫”在同一台主机上的繁殖次数，但是，这位天才编制的计数器竟然有一处小小的疏忽。

@本来设定只繁殖一次，可结果是“蠕虫”爬遍互联网络，过度繁殖耗尽了众多主机的系统资源，基本“当”掉了整个Internet。这一切就发生在1988年11月2日晚。

@莫里斯释放了这条“小虫”后回到宿舍安然入睡的几个小时之内，就引发了一场对当时约有6万台电脑的互联网的毁灭性攻击。他让人们彻底认识到了Internet的弱不禁风。

68

蠕虫之父？

@是否可以说Robert Morris就是“蠕虫之父”呢？真正的“蠕虫之父”应该是John Shoch。

@John Shoch当时设计了一个程序：网络空闲时，该程序就在各计算机之间“游荡”而不会造成任何破坏；当有机负荷过重时，该程序可以从空闲计算机“借取资源”，达到整个网络的负载平衡。

@莫里斯的蠕虫程序就是受此启发而设计的，只不过是“借取空闲资源”改为“耗尽所有资源”。

70

Morris 蠕虫主要攻击方法

@主要的攻击方法

@Rsh, rexec: 用户的缺省认证

@Sendmail 的debug模式

@Finger的缓冲区溢出

@口令猜测



72

Morris Worm的诞生

@损失是巨大的，直接经济损失上百亿美元，仅就上万名工程师夜以继日地恢复工作来计算就是一笔不小的支出。

@1990年，Robert Morris成了首位依据美国反黑客法律《计算机欺诈和滥用法》被判决的人。

@专家们分析后得出结论：Robert Morris确实不是故意的。据此他被从轻发落：罚款一万美元，三年徒刑，400小时社区劳动。

@而今 Robert Morris在麻省理工学院任教。

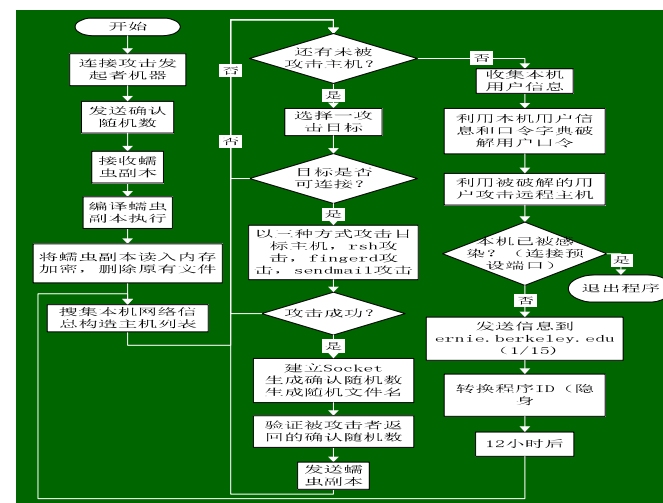
69

蠕虫之父？

@当莫里斯事件发生时，John Shoch说：“它是我们首先创造出来的，但我们没有很好地控制它”。莫里斯只是从反面对其进行了一次代价不菲的演习而已。

@那一年，美国总统里根因此而签署了《计算机安全法令》。

71



73

CodeRed

- @红色代码病毒是一种结合了病毒、木马、DDOS机制的蠕虫。
- @2001年7月中旬，在美国等地大规模蔓延，开创了病毒史的新纪元。
- @红色代码病毒利用微软IIS的安全漏洞进行病毒的传播。
- @2001年8月初，出现变种codered II，针对中文版windows系统，国内大规模蔓延。
- @通过80端口传播。只存在与网络服务器的内存，不通过文件载体。

74

CodeRed II

- @增加了特洛伊木马的功能，并针对中国网站做了改进。
- @拷贝系统目录cmd.exe到IIS的脚本执行目录下，改名为root.exe
- @将病毒体内的木马解压缩写到C盘和D盘的explorer.exe
- @木马每次系统和启动都会运行，禁止系统的文件保护功能，并将C盘和D盘通过web服务器共享
- @计算IP的方法进行了修改，使病毒传染的更快。
- @检查是否存在CodeRedII原子，若存在则进入睡眠状态防止反复感染，若不存在则创建CodeRedII原子。
- @创建300个线程进行传染，若系统默认语言为简体中文或繁体中文，则创建600个线程。

76

CodeRed I

- @在侵入一台服务器后，其运行步骤是：
- @设置运行环境，修改堆栈指针，设置堆栈大小为218字节。接着使用相对虚拟地址RVA查找GetProcAddress的函数地址，获得其他socket、connect、send、recv、closesocket等函数地址；
- @如果C:\notworm存在，不再进一步传染；
- @传染其他主机：创造100个线程，其中99个用户感染其他WEB服务器，被攻击IP通过一个算法计算得出。
- @篡改主页，如果系统默认语言为“美国英语”，第100个进程就将这台服务的主页改成“Welcome to http://www.worm.com !, Hacked By Chinese!”，并持续10个小时。
- @如果时间在20:00UTC和23:59UTC之间，将反复和白宫主页建立连接，并发送98k字节数据，形成DDOS攻击。

75

CodeRed II



@攻击形式

- @http://x.x.x.x/c/inetpub/scripts/root.exe?/c+dir
- @http://x.x.x.x/c/winnt/system32/cmd.exe?/c+dir
- @其中x.x.x.x是被攻击的IP地址，dir可以是任意命令，比如删除系统中的文件，向外发送机密数据等，这个后门后来也成为了Nimda病毒的一个传播模式。

77